This is to certify that the
thesis entitled

MECHANISMS FOR COMBINING INFRARED AND
ULTRASOUND SIGNALS FOR INDOOR WIRELESS
LOCALIZATION

presented by

DAVID RURANGIRWA

has been accepted towards fulfillment
of the requirements for the

M.S. degree in ELECTRICAL AND COMPUTER ENGINEERING

Major Professor's Signature

11/30/07

Date

MECHANISMS FOR COMBINING INFRARED AND ULTRASOUND SIGNALS
FOR
INDOOR WIRELESS LOCALIZATION

By

David Rurangirwa

A THESIS

Submitted to
Michigan State University
In partial fulfillment of the requirements
For the degree of

MASTER OF SCIENCE

Electrical and Computer Engineering

2007

ABSTRACT

MECHANISMS FOR COMBINING INFRARED AND ULTRASOUND SIGNALS FOR INDOOR WIRELESS LOCALIZATION

By

David Rurangirwa

This thesis presents a new mechanism for secure indoor localization through a combination of infrared and ultrasound signals. While a number of existing systems use ultrasound and radio frequency (RF) based localization, the use of RF gives rise to a series of operational difficulties including lack of localization privacy and collisions among the localization beacons. In this thesis, infrared is used to mitigate these limitations of RF. Collisions among the localization beacons, placed in different rooms, are avoided by leveraging the attenuation of infrared signals through walls and other indoor partition materials. Privacy is ensured by the complete isolation of the infrared signal across different rooms and hallways. Also, the unlicensed usage of infrared can provide a significant operational advantage compared to the RF based solutions.

We implement a *time difference of arrival* (TDOA) mechanism in which the localization beacons send simultaneous infrared and ultrasound pulses which are received at localization modules, which compute the distance to a beacon by measuring the TDOA between the IR and the US signals. Applications of such indoor localization systems include robot navigation, location-aware sensor network protocols, equipment localization, and various location-based wireless services.

To My Mother

For All Her

Love, Prayers, and Support

Acknowledgements

I am grateful to God for good health, without which I could not have accomplished much. I am also grateful to Fulbright for sponsoring my education at Michigan State University.

At the start of the second year of my master's program, I asked Dr. Biswas to direct my master's thesis research. I explained to him that I wanted to develop a network system. He then suggested to me the indoor location system based on infrared and ultrasound. Throughout this work, Dr. Biswas has been of great help, always finding time to brainstorm, troubleshoot, and make recommendations, without which, this work could not have been possible. I am especially thankful for his systematic approach to problem solving which enabled me to complete this project. I would like to sincerely thank him for not only supporting me in my research, but also for his advice, motivation, and his willingness to let me decide which way I should go.

Fan Yu worked with me from the start of the project until the final day when the whole system was working. Thanks to him, I have acquired more skills of system debugging. I would like to sincerely thank him for his tireless efforts to see that this master's work was complete. All along, he kept an easy outlook of things, which I am also very thankful for.

I would also like to thank Jayanthi Rao for her help in debugging the system's code. She was always eager to help and was happy when things seemed to fall into place.

My fellow students in the NEEWS lab created an environment conducive to research and I would therefore like to thank them. Their questions and suggestions during the demo of this project were of great help.

I would also like to thank the ECE shop staff for their great help in acquisition of system components and experimental apparatus. I made numerous visits to the shop and every time they were eager to help me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1: **Introduction**

This chapter gives the background and motivation for this thesis work. An effort is made to lay the foundation for consequent topics. It also describes the contributions that this thesis work has made.

1.1 Background

Localization is a mechanism that is used to remotely find the location of objects, either indoors or outdoors. It has been a challenge for researchers to devise means of telling the exact location of men and their belongings. Up to date, there continues to be tremendous breakthroughs into this field. Currently, through the Global Positioning System (GPS), it is possible to locate a person within a few meters of accuracy. This technology has found many applications such as vehicle navigation and tracking of people and animals.

The GPS and RADAR systems have been employed to solve user location and tracking in outdoor scenarios. However, it is becoming more and more evident that indoor location and tracking systems are needed for numerous applications. The conventional means of outdoor location cannot be employed for indoor purposes due to obstacles found inside buildings, which reflect and attenuate data signals [1]. It is therefore necessary that systems that are specifically suited for indoor localization be developed. Some of the applications of indoor location systems are:

- Localizing visually impaired navigators

- Navigation tools for humans and robots
- Finding critical personnel or resources faster
- Asset tracking
- Location-aware sensor networking
- Simplifying the user interface for mobile voice and data functions
- Improving the security and effectiveness of Wi-Fi networks
- Restricting online shopping to certain people in a certain room
- Improvement of roaming capabilities.

There have been many solutions suggested for indoor localization, some of which are: In-building RADAR [5], Active Bat Location system [10], Active Badge Location System [3], HiBall Head Tracking system [6], Ubisense Location system [11], Broadband Ultrasonic location system [2], MIT Cricket [1], Bristol Indoor Position system [7], and Nibble location system [9].

1.1.1 Indoor Location Systems

Indoor localization involves objects finding their position in reference to other objects in an indoor scenario. The reference objects may be stationary or mobile. The object seeking to know its position and the reference object may be active or passive. In case they both have to be active, some synchronization and scheduling mechanisms are needed. In many cases, the object seeking to know its position is passively listening to messages sent from the reference point. In this case, the synchronization step is eliminated, but scheduling has to be implemented, especially if there has to be multiple transmitters.

1.1.2 *Indoor Location System Application Example*

Figure 1 shows a possible application of the IRUS indoor location system. The robot can be programmed to move within the boundaries demarcated by beacons 1 & 2. For instance, the beacons can be used to represent edges of a table or any elevated surface from which the robot should not topple and fall. Simply put, the robot sees its boundaries.

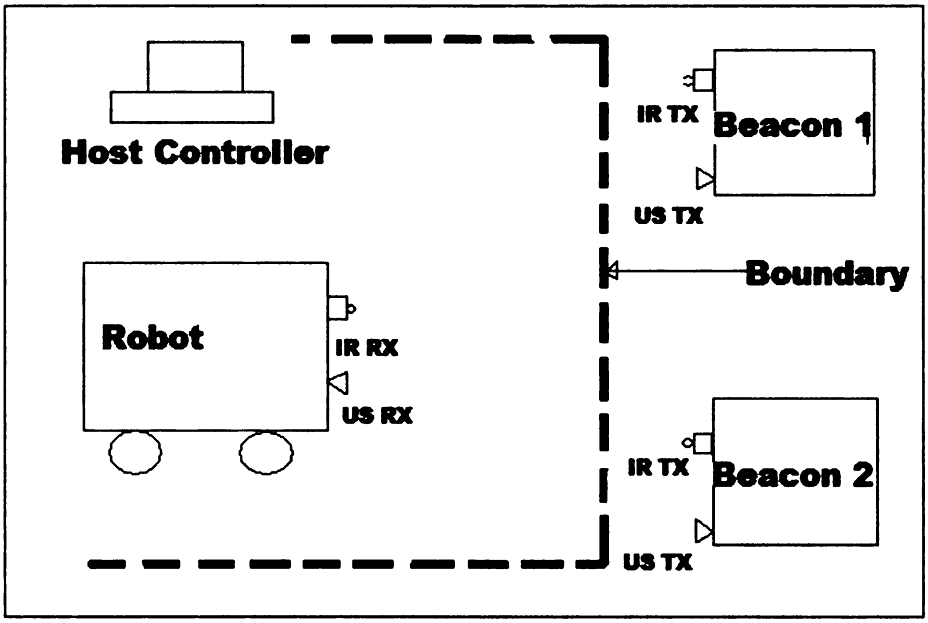


Figure 1: Example of an indoor localization application

The indoor localization devices can be distinguished by the media used for data exchange and the algorithms used to achieve localization. The media that have been explored so far are Radio Frequency, Infrared, and Ultrasound. Some of the algorithms used are: Angle of Arrival (AOA) or Direction of Arrival (DOA) [12], Time of Arrival

(TOA) [10], and Time Difference of Arrival (TDOA) [1].

The indoor localization solution presented in this thesis work is based on time difference of arrival (TDOA) of infrared and ultrasound signals. There are two modules in the system designed; a transmitter (beacon) and a receiver (listener). The beacon sends an infrared signal followed by an ultrasound signal. The listener detects these signals and computes the difference in their times of arrival. This difference is then multiplied by the speed of sound to obtain the distance separating the two modules.

Time difference of arrival has been employed in the MIT Cricket system [1,2]. This system uses RF and US signals to achieve distance computation based on TDOA. We introduce the use of infrared because of its inability to penetrate walls, therefore reducing interferences to neighboring rooms.

1.2 Motivation

Indoor environments have different characteristics. To design an indoor location system, this factor should be considered. We had this consideration in mind when designing the system in this thesis. We wanted to explore an alternative medium of data transmission, other than RF. IR was opted for because of its inability to penetrate walls. This feature would ensure protection of user privacy and avoid interferences to neighboring rooms.

1.3 Contributions of the Thesis

In this thesis, a new method of indoor localization is presented. This method involves periodically sending an infrared signal followed by an ultrasound signal. The receiver of these signals (listener) records their times of arrival and based on the time difference of arrival (TDOA) [1] algorithm running on it, it calculates the distance separating it and the signal sender (beacon).

We developed a prototype of the indoor location system using off-the shelf components. We then carried out experiments to characterize and evaluate the performance of the system under different indoor conditions including ambient lighting and temperature.

1.4 Thesis Organization

This chapter of the thesis presented a background of indoor location systems. The next chapter is a survey of related work in this area of networking. Chapter three gives a detailed description of the concept behind indoor localization presented in this thesis. The fourth chapter presents a detailed description of the hardware as well as the software architecture of the indoor location system designed in this thesis. Chapter five presents the characterization procedures, followed by the results. It also outlines the challenges encountered and how they were overcome. The sixth chapter presents the conclusion of this work and the seventh chapter presents recommendations for future work.

Chapter 2: **Related Work**

This chapter describes some of the existing indoor location systems. Research on indoor location systems is a fast growing area and this chapter is in no way exhaustive as far as their coverage is concerned. However, an attempt is made to cover as much a wide area as possible, in this dynamic research area.

2.1 The MIT Cricket

The MIT Cricket [1, 2] is an indoor location system based on time difference of arrival of radio frequency and ultrasound signals. It is composed of beacons and listeners. Beacons are strategically placed, either on ceilings or on walls. They periodically transmit RF signals with id information. At the same time, an ultrasonic signal is transmitted. At the listener, the RF signal is received first and its time of arrival noted. The ultrasound signal lags behind the RF signal, and this lag period is determined at the listener. It is then multiplied by the speed of sound to calculate the distance between the beacon and the listener.

The Cricket system is distributed and scales well since the listeners are passive. However, there is a likelihood of RF signals leaking to neighboring rooms. The system suggested in this thesis is similar to the cricket system in terms of distance determination. However, instead of RF, we use infrared. The advantages that arise from this change were discussed in section 1.2.

2.2 Broadband Ultrasonic System

This system, developed at the University of Cambridge, can be seen as a way to enhance the Cricket system by increasing the transmission rate in the Ultrasound channel. Unlike the Cricket system and the ranging system in this thesis work, the Broadband Ultrasonic system incorporates data into the ultrasonic signal.

The Broadband Ultrasonic location System is polled and centralized [2]. When a system is termed as polled, it means that the transmission is coordinated. The centralization property means that a centralized system exists to collect and analyze data from the nodes.

To determine the location of a node, times of flight of messages between transmitters and receivers are used to compute the distances between the transmitter and receiver by multiplying times of flight by the speed of sound in air.

The weakness of the Broadband Ultrasonic System can be seen as a lack of user privacy due to the use of a centralized system to collect and analyze data; and the huge amount of energy used to achieve the broadband data transmission.

2.3 The Active Badge Location System

The Active Badge Location System [3] is based on badges that transmit signals with information about their location to a centralized system. A sensors network is put around a host building, and it picks up signals that are periodically transmitted. These badges are worn by people in an area, whose locations need to be determined. The signals transmitted are infrared signals.

The drawback of the Active Badge is its user privacy compromise since it is based on a centralized system.

2.4 The Bat Ultrasonic Location System

In order to improve the Active Badge system, the Bat Ultrasonic Location System [4] was developed. Unlike the Active Badge System, this location system is able to provide location and orientation information in 3D. To obtain location information, times of flight of an ultrasonic signal between a transmitter (Bat) and the object to be located; are determined and these are multiplied by the speed of sound to calculate the distances from the Bat to each receiver. With three or more bat-receiver distances, enough information is available to determine the Bat's 3D position. The orientation of an object can be calculated by determining the relative positions of two or more Bats that are attached to that object.

The disadvantages of relying on ultrasound to transmit data still apply to the Bat Ultrasonic Location System.

2.5 In-Building RADAR

This indoor location and tracking system is RF-based [5]. Information describing signal strength is theoretically computed and empirically-determined at multiple receiver locations. This information is then used to triangulate the coordinates of the user. This system suffers from effects of radio channel interferences, which reduce its accuracy.

2.6 The HiBall Tracking System

This electro-optical system was built for high precision head tracking for virtual reality applications [6]. It consists of panels of LED's that are flashed sequentially, head-mounted cameras that determine the position of the flashing LED's, and a computer system that uses the knowledge about the cameras' coordinates to obtain the location information. The drawback of this system is its expensive implementation.

2.7 Low Cost Indoor Positioning System

This system, developed at the University of Bristol-UK, is based on radio frequency and ultrasonic signals [7]. The RF signal is used to synchronize the transmitters and the receiver. Four ultrasonic transmitters are placed on the ceiling of the experimental room. The signals sent are detected and their times of flight are recorded at the ultrasound receiver. The times of flight are factored with the speed of sound to determine the distances between the transmitters and the receiver. Four times of flight are used to ensure that the system's range is increased and that signals that were lost are compensated.

The synchronization mechanism used in the low cost indoor positioning system increases the cost of the system. Since both the transmitter and receiver have to be active, this system fails to scale very well.

2.8 The Horus WLAN Location Determination System

The Horus location system [8] is based on RF. Signal strengths of frames transmitted by the access points are used to provide user location. The system is currently implemented in the 802.11 wireless LANs context. Since it is based on the already existing wireless LANs, it is seen as a software solution built on top of the wireless infrastructure. Just like other WLAN technology location systems, Horus works in two phases. The first phase is referred to as *offline* phase. In this phase, data representing signal strength is collected from points of access and tabulated into a radio-map. The second phase is called the location determination (*online*) phase. Here, the system searches the radio-map based on newly received signals from the access points to estimate the user location.

The drawback of the Horus WLAN system is in the fact that there can be interferences from neighboring rooms or other RF based devices, which would result to incorrect samples in the *offline* phase.

Chapter 3: **Concept**

This chapter presents the concept behind the system designed in this thesis. It explains the principle behind ranging and distance computation based on time difference of arrival (TDOA) of infrared and ultrasound signals.

3.1 Ranging

Indoor location systems have ranging as their core function. Ranging involves determining the distance separating one point from another. To achieve this, various techniques can be employed.

Most organisms are cognizant of their surroundings through eyesight. Eyesight can judge distances and tell the organism how far it should fly, run, fall, and so on. Eyesight is in essence a ranging mechanism based on reflection of light off of objects to the retina of the eye. The brain then processes this information and the organism is able to interpret it and is therefore able to navigate its surroundings. Some indoor localization systems mimic the eyesight mechanism. They send a light signal e.g. infrared, which bounces off of an object. The reflection is then interpreted by a microprocessor of the receiving device. The information extracted can tell a robot, for instance, how far or near a boundary it should navigate.

The bat is a blind, flying mammal and its ranging mechanism has inspired many commendable discoveries. It sends out ultrasonic signals that are reflected off of its surroundings and fall on its ears. Its brain then processes this information and it is able to judge its surroundings at amazing precisions. This idea has been employed in ranging systems. For instance, ultrasound based ranging systems use echoes to calculate distances. A node seeking to know its position sends out an ultrasound signal which is reflected off a reference point. The transmitting node has to record the time that the signal was sent. When the signal reaches the reference point, it is reflected back to the transmitter, which records the time of arrival of the reflected signal. The difference in the departure and arrival times corresponds to the distance separating the transmitter and the reference point. This difference is then multiplied by the speed of sound to obtain the distance separating the node and the reference point. Figure 2 represents ultrasonic ranging.

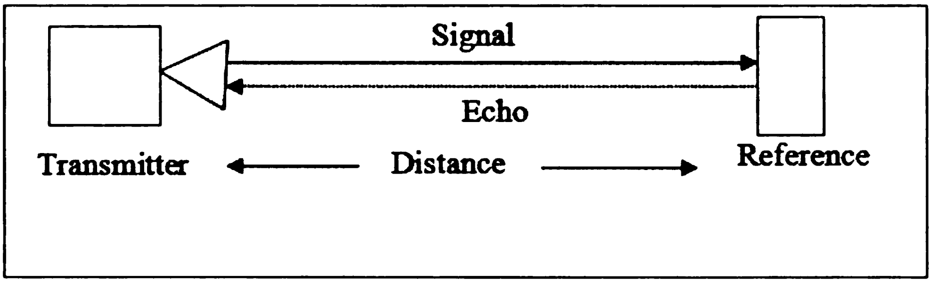


Figure 2: Ultrasonic Ranging

If there are two systems that have a synchronized clock, either of them can tell the distance from the other by noting the time of departure and arrival, and then multiplying the difference by the speed of the medium of propagation. The transmitter has to send the packet with time of departure information. The receiver will then extract this information and read its clock for the time of arrival.

The preceding two methods have their strong as well as weak points. The echo/reflection based ranging is simple and inexpensive but suffers from interferences. For instance, any object that can reflect the signal used can be confused for the reference point, thus leading to erroneous computations of distances. The second method that involves synchronization ensures that distances are computed only at the correct reference points, but it is costly and has a limitation as far as scaling is concerned.

Suppose two signals moving at different speeds left a transmitter at the same time. They would reach a reference point at different times. The reference point would then take note of this difference in times of arrival and use them to calculate the distance from the transmitter. This method of ranging is based on a technique called "Time Difference of Arrival"-TDOA [1]. TDOA avoids the necessity to synchronize the system clocks, and can therefore be less costly than the synchronized system. It also scales well since the transmitter is active but the listener is passive. Section 3.2 gives a detailed description of this method of ranging.

3.2 Ranging based on TDOA of IR and US Signals

The solution suggested in this thesis is based on ranging using the TDOA algorithm [1], based on infrared and ultrasound signals. The TDOA technique has been used in the MIT cricket [1, 2], where radio-frequency was used together with ultrasound. We propose the use of IR due to its following advantages:

- Minimal interference to/from other rooms, since IR transmissions cannot penetrate walls. This feature also provides user privacy.
- Unlicensed data transmission, allowing for flexibility of experimentation and prototyping
- No requirement for an antenna which is a cost and size issue in RF technologies

In the system described in this thesis, ranging is achieved using TDOA of infrared and ultrasound signals. The beacon periodically transmits an infrared signal followed by an ultrasound signal. These signals are detected at the listener. The speed of light and that of sound are known. These values can then be used, together with the times of arrival, to determine the distance between the beacon and the listener.

Let the speed of IR be V_{ir} and that of sound be V_{us}. Since V_{ir} is greater than V_{us}, the ultrasound signal lags behind the infrared signal as they move from beacon to listener. The listener determines the time lag, δT, and can calculate the distance D, from:

$$\delta T = \frac{\frac{D}{1}}{V_{us}} - \frac{\frac{D}{1}}{V_{ir}} \quad (1)\ [1]$$

Under normal conditions, the speed of sound is approximately 344m/s, and that of light is 3×10^8m/s. Since $V_{ir} >> V_{us}$,

$$D \approx \delta T.\ V_{us} \quad (2)\ [1]$$

3.3 Factors Affecting Distance Computation based on TDOA of IR and US signals

As shown in equation 2, distance computation depends on the accurate determination of δT as well as the speed of sound. The accurate determination of δT in turn depends on the accuracy of the time recording algorithms running on the listener, as well as the distance computing algorithms.

On the other hand, the speed of sound depends on environmental factors such as relative humidity, temperature, and atmospheric pressure. For example, At 25 ^{0}C and 101.325 kPa (atmospheric pressure at sea level), the speed of sound changes by only about 0.5% as relative humidity changes from 0% to 100% [1]. At 25 ^{0}C & 50% relative humidity, the speed of sound changes by only about 0.6% as the atmospheric pressure changes from 101.325 kPa to 30 kPa (atmospheric pressure at the top of Mount Everest) [1]. Therefore, atmospheric pressure and relative humidity have limited effect on the speed of sound.

In air, the speed of sound changes by 0.18% for every 1 ^{0}C change at 25^{0}C [1]. This change is approximately equal to 60cm/s change in the speed of sound for every 1 ^{0}C change in temperature. This property necessitates compensation. To monitor temperature change and adjust the speed of sound appropriately, we used a temperature sensor. For experimental purposes only, the temperature sensor is only included in the listener circuitry.

3.4 Node Localization Mechanisms

To find the exact location of objects, we would need several nodes to communicate. For example, figure 3 shows node localization based on coordinate axes. The axes can represent walls, floor, or ceiling of a room. Figure 4 shows localization based on inter-node distances. All the nodes can have information about distances to neighboring nodes. Distances from three nodes can give user location as well as orientation information.

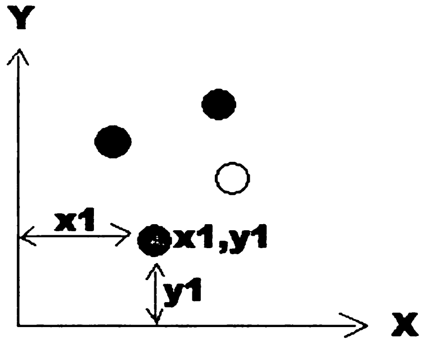


Figure 3: Node coordinate assignment from coordinate axes

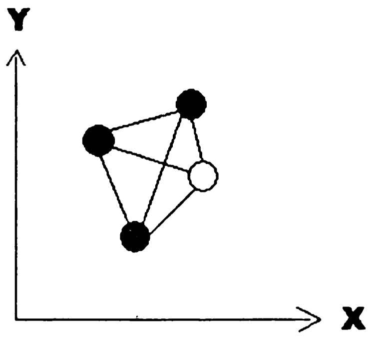


Figure 4: Node coordinate assignment from inter-node distances

Chapter 4: **System Architecture**

This chapter outlines the hardware as well as the software architectures of the IRUS wireless indoor location system. It also gives a detailed analysis and description of the system's configuration and parameters.

Figure 5 shows the modules of the IRUS indoor location system designed in this thesis and figure 6 is a depiction of its organization. The Beacon [1] which is the transmitting node can be located on the ceiling, elevated points in a room or on the wall while the listener [1], which is the receiving node is attached to a host PC or device that needs to be localized.

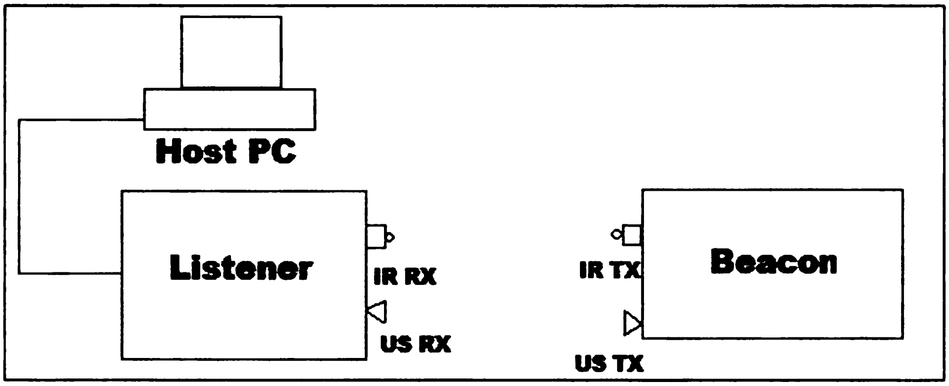


Figure 5: Modules of the IRUS Indoor Location System

Since the listener is passive, this system can scale very well and also ensures that user privacy is protected. Both infrared and ultrasound do not penetrate walls. Therefore, interferences to/from neighboring rooms are avoided.

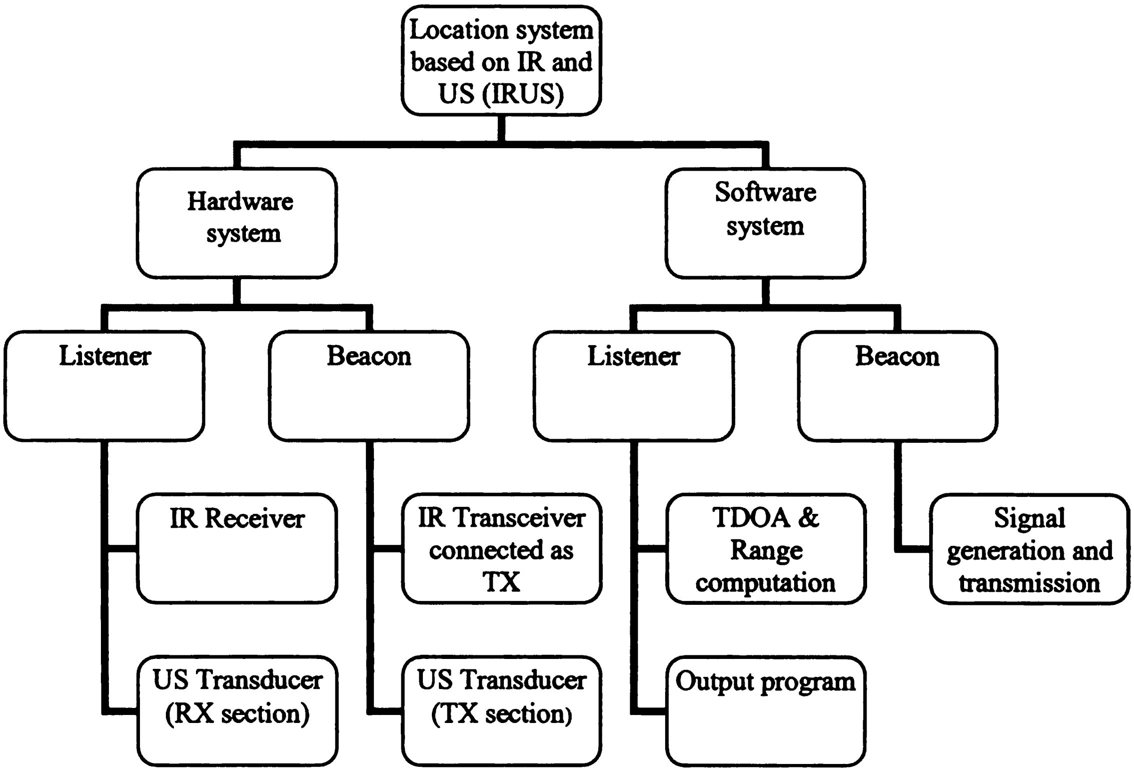


Figure 6: Components of the IRUS indoor location system

4.1 Hardware Configuration

This thesis work was hardware intensive. Therefore, a significant amount of time was spent on the hardware design. The system was assembled on a proto-board, using off-the shelf components.

4.1.1 Overview of System

Figures 7 and 8 show block diagrams of the listener and beacon used in the indoor location system based on Infrared and Ultrasound signals. Each module consists of a microcontroller, ultrasound transducer and supporting circuits, and infrared devices.

The beacon has the TFDU4300 IR transceiver configured as an IR transmitter, while the listener has the TSOP381 for IR signal reception. The infrared transceiver is connected to the microcontroller via an external UART. The external UART used is MAX3100.

The microcontroller used is PIC16F874, and runs at 20 MHz. For lower power consumption, a slower clock can be used. This microcontroller was picked due to its simplicity and inexpensiveness. It is based on the RISC architecture and has only 35 single word instructions.

The Ultrasound transducer used is similar to the one used in the MIT Cricket [1, 2]. Modifications were made to the supporting circuitry to suit our application. See schematic in appendix A.

To communicate with the host PC, the listener runs the RS232 protocol, taken care of by the RS232 chip. The listener sends data to the host at a baud rate of 9600bps.

Only the listener has a temperature sensor. This option was taken so as to simplify the design of the experimental system. The temperature sensor will be incorporated in the beacon architecture once the beacon has been programmed to send an IR message. The temperature sensor used is LM34DZ, which is a precision Fahrenheit temperature sensor. Its output voltage is linearly proportional to the temperature in Fahrenheit. This temperature sensor was configured as a basic Fahrenheit temperature sensor and calibrated to measure temperatures in indoor scenarios as low as 50^0F and as high as 89^0F.

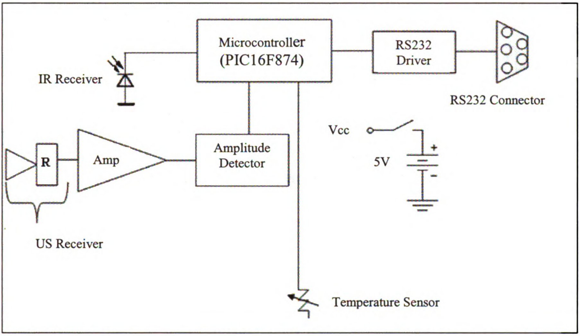


Figure 7: Block diagram of listener [1]

For experimental purposes, only two modules, one beacon and one listener, were built. Therefore, since there was no risk of sending a signal from the wrong beacon, the listener was not programmed to process the message sent by the beacon. Every signal from the beacon is simply taken as a pulse that triggers an interrupt at the listener.

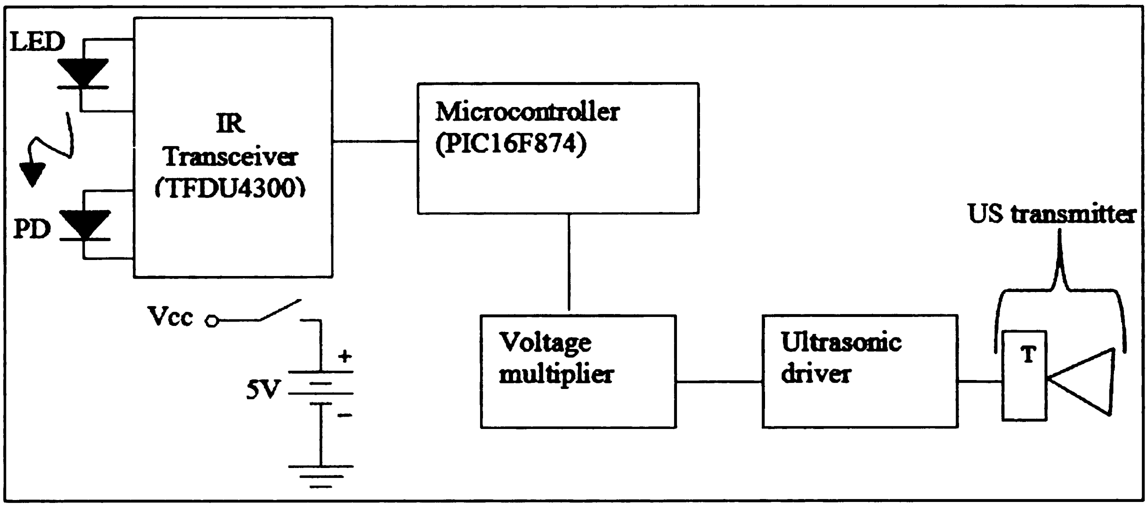


Figure 8: Block diagram of beacon [1]

4.1.2 IR transceivers

The TFDU4300 Infrared transceiver was used as an IR transmitter at the beacon. Figure 9 shows the beacon's infrared hardware section. The microcontroller generates signals to configure the external UART to transmit infrared data according to IrDA standard at a data rate of 9600 bps. The external UART (MAX3100) receives data from the Serial Peripheral Interface (SPI), formats it to IrDA, and sends it to the Infrared transceiver, which in turn transmits the infrared signal to the infrared receiver on the listener side. MAX3100 was designed to directly drive optocouplers, whereas IrDA modules have inverting buffers. This feature calls for inversion of the TX and RX signals. A NAND gate was used for this purpose.

TFDU4300 is compliant to the IrDA standard and can also be configured to transmit data in the remote control mode. The transmitter has an output radiant intensity equal to 65 mW/sr and a peak emission wavelength of 880-900 nm. Its spectral bandwidth is 45 nm. It has an optical rise and fall time of 10-100 ns. For an input pulse width of 1.63 μs, the optical output pulse duration is 1.6-1.8 μs. For an input pulse width greater than or equal to 20 μs, the optical output pulse duration is 20-300 μs. In all these cases, a data rate of 115.2 kbps is assumed [33].

For short distance transmissions, TFDU4300 can also be used as a receiver. In SIR mode the receiver has a minimum detection threshold irradiance of 40-80 mW/m2. Its maximum detection threshold irradiance is 5 kW/m^2. The rise and fall times of the output signal are 10-100 ns. The Rxd pulse width of the output signal for an input pulse length greater than 1.2 μs is 1.65-3.0 μs. Its stochastic jitter at the leading edge is 250 ns at a data rate less than or equal to 115.2 kbps [33].

For long distance transmissions, TSOP341 IR receiver module was used. Its typical transmission distance is 45 m. Its minimum irradiance is 0.1-0.25 mW/m^2, while its maximum irradiance is 30 W/m^2. Its directivity is $\pm 45^0$ [32].

4.1.3 US transducers and Supporting Circuits

At the beacon, the microcontroller produces a periodic wave, at 40 KHz through its pulse width modulation mode (PWM). This signal is then amplified so as to drive the ultrasound transducer. The ultrasound transducer used is the 255-400 series where the transmitter is 255-400ST12 and the receiver is 255-400SR12. It has a center frequency of 40 KHz $\pm$1.0 KHz. The sensitivity of the receiver at the center frequency is -67dB. The minimum driving voltage is 1V, while the maximum is 20V. The sensitivity of the receiver at the center frequency is -67dB. The minimum driving voltage is 1V, while the maximum is 20V. The transducer's bandwidth is 2 kHz. [30].

Typically, the ultrasound signal that triggers an interrupt at the listener is approximately 5V. The amplification of the ultrasound signal to this level is done at the listener. The amplifier circuit has a potentiometer that can be varied manually to adjust the sensitivity level of the circuit to the incoming ultrasound signal. This potentiometer can be adjusted for short distance range computations. See schematic in appendix A.

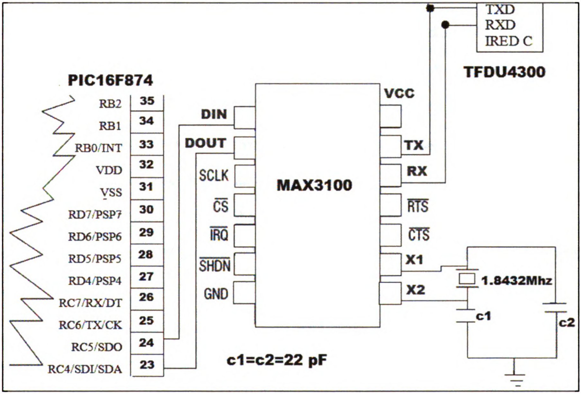


Figure 9: Infrared Section of the beacon

4.1.4 Microcontroller

PIC16F874, the microcontroller used in the design of the IRUS location system, is an 8-bit microcontroller. It has up to 8K x 14 words of FLASH Program Memory, up to 368 x 8 bytes of Data Memory (RAM), and up to 256 x 8 bytes of EEPROM Data Memory. It is able to handle up to 14 sources of interrupts. Its hardware stack is 8 levels deep. It supports direct, indirect and relative addressing modes. It features a power saving SLEEP mode. Its operating voltage range is 2.0 V-5.5 V. Power consumption can be as low as <0.6mA, operating at 3 V and 4 MHz. Operating at 3 V and 32 kHz, a 20 µA low power consumption is feasible. The typical standby current is <1 µA. [34, 35]

The peripheral features of PIC16F874 were exploited to achieve the system functionalities of the indoor location system described in this thesis.

On the beacon side, timer2, an 8-bit timer with an 8-bit period register, prescaler and postscaler; was used to control the PWM module. The resolution of PWM can be as high as 10-bits. The PWM mode enables the generation of pulses of various periods and duty cycles.

The Synchronous Serial Port (SSP) with Serial Peripheral Interface (SPI) was used in the master mode to generate serial data that was converted to IrDA by an external UART, MAX3100, thus making transmission of infrared signals possible.

On the listener side, timer1, a 16-bit timer/counter with a prescaler, was used together with an external crystal; to keep time of the system. This timer is the one that enables recording of the times of arrival of signals to be done. Reading timer1 takes place in two steps: reading the high byte and reading the low byte. In order for the time information to be correct, reading must take place before or after rollovers.

Readings between rollovers lead to erroneous computations. At 20 MHz, a 16-bit counter will roll over every 13ms. This property calls for the implementation of a rollover counter that should also be read every time timer1 is read. A 32.768 kHz crystal oscillator was used to control the timer1 as a timer in asynchronous mode. In this configuration, the microcontroller can actually keep real time and the need of a rollover counter is avoided. However, since the timer is read in two stages, it is still possible to read a value between rollovers of the lower byte of the timer. This error is checked in software. Section 3.2.2 gives a detailed description of how this is achieved.

The microcontroller also features an analog-to-digital converter (ADC) which is 10-bit and is multi-channel. This feature is the one that was employed for temperature sensing. The output of the temperature sensor is connected to one of the ADC input pins and the voltage value present on the ADC pin is converted to a digital value, which is then used to represent the ambient temperature in the lookup table implemented in software.

The Universal Synchronous Asynchronous Receiver Transmitter (USART) was used to send data to the host PC. It was configured as an asynchronous, high speed transmitter, transmitting data at a baud rate of 9600 bps.

PIC16F874 supports In-Circuit Serial Programming (ICSP). This mode of programming was used to download programs into the microcontroller. The programmer used was the MPLAB In-Circuit Debugger (ICD 2).

4.1.5 System parameters

This section outlines the parameter values of the various components of the IRUS wireless indoor location system.

4.1.5.1 Beacon

Table 1 shows the parameters of the transmitting module (beacon). Here, beacon frequency refers to how often the beacon transmits signals to the listener and it is shown to be 1 second. The maximum clock speed was used in the prototype, but a lower speed can be used for the clock and this can result to lower power consumption. The IrDA baud rate used is the rate that showed best compliance with the external UART (MAX3100). The ultrasound pulse duration used was obtained through experimentation and it gave the best performance for most distance values.

Parameter	**Value**	**Description**
Beacon Frequency	1 Hz	Beacon sends signals every second
Microcontroller clock speed	20 MHz	Maximum clock speed, resulting to 200ns instruction cycle
IrDA baud rate	9600 bps	Standard IrDA data rate
IR wavelength	880-900ns	Typical IrDA wavelength
US frequency	40 KHz	Optimum response frequency of the transducer used
US Pulse duration	1500 μs	Optimum duration for proper US signal detection at the listener

Table 1: Beacon parameters

4.1.5.2 Listener

The listener parameters are shown in table 2. The clock speed and the US transducer frequency are similar to those of the beacon. The listener features a precision temperature sensor that is able to measure temperatures to ±1°F. Temperatures likely to be measured in an average indoor location were considered when calibrating the temperature sensing mechanism of the listener. This feature can be readily modified to cover the full range of temperatures.

Parameter	**Value**	**Description**
Microcontroller clock speed	20 MHz	Maximum clock speed, resulting to 200ns instruction cycle
US frequency	40 kHz	Optimum response frequency of the transducer used
US RX gain	70-78 db	Detection of the US signal at approx. 8 m when the modules are directly opposite each other [1].
Temperature sensor range	50-89 ^{0}F	Range of temperatures that are likely to be found in indoor environments
RS232 data rate	9600 bps	Standard data rate

Table 2: Listener parameters

4.2 Software Configuration

The programs that run on the beacon and listener were written in assembly language, and tested using the MPLAB IDE. Codes were written, and compiled using the MPASM feature of the MPLAB IDE. Simulation followed using the MPLAB simulator. For supported functions, external stimuli were introduced to simulate system performance. However, some features of the microcontroller cannot be simulated. For instance, currently, the real-time clock mechanism, serial-peripheral interface, cannot be simulated, thus the code for those functions had to be tested on the physical system. After the necessary simulations and debugging, the resulting hex files were downloaded into the microcontrollers.

Figure 10 represents a summary of the initializations done in software for both the beacon and listener. Sections 4.2.1 and 4.2.2 give a more detailed description of what goes on inside the microcontrollers of the beacon and the listener.

4.2.1 Beacon Program

The beacon runs a program that periodically sends an infrared signal followed by an ultrasound signal. The initialization part of the code involves configuring the microcontroller to send serial data via the Serial Peripheral Interface port to the external UART. The microcontroller also generates the configuration word for the external UART. The external UART converts the serial data to IrDA data, which is then passed on to the IR transmitter.

Initialization also involves enabling the pulse width modulation mode and produces a wave of approximately 40 kHz for 1500 μs, which is the Ultrasound signal. To control the duration of the ultrasound signal, the duty cycle is defined as 50% for 1500 μs and then it is defined as 0% for the duration within which the infrared signal is generated.

The repetitive loop of the beacon program involves calling for the serial data signal to be transmitted as infrared, followed by the ultrasound signal. This sequence is repeated every second.

The initialization routine takes approximately 17.6 μs. The infrared signal generation routine takes 9.4 μs, after which the ultrasound signal is generated. This delay is very negligible and it is therefore assumed that the signals leave the beacon at the same time.

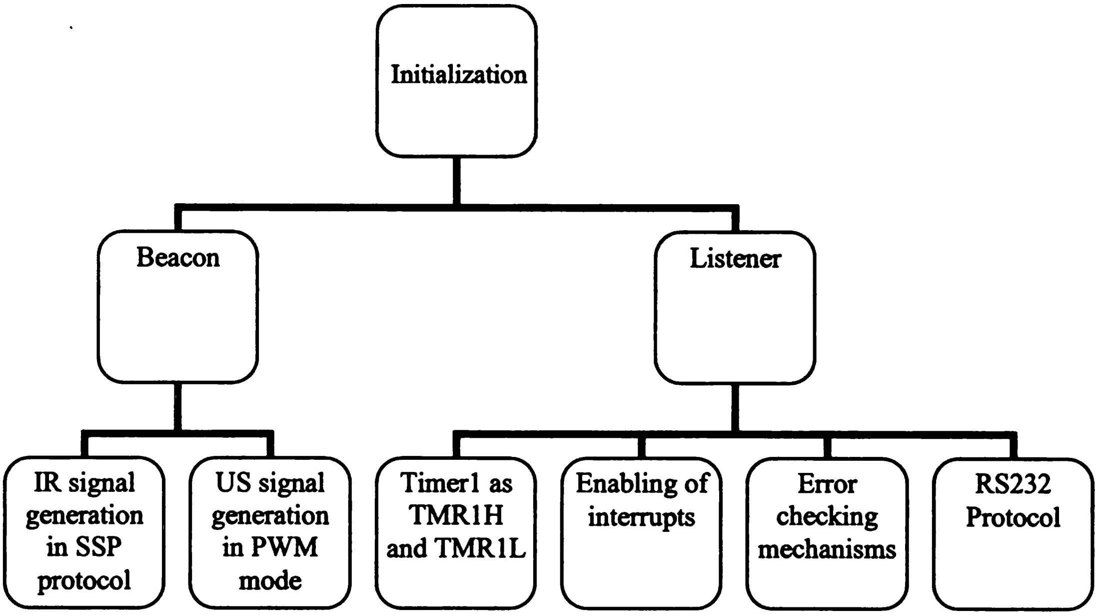


Figure 10: System Initializations

4.2.2 *Listener Program*

The listener runs a program that detects the signals transmitted from the beacon; records their times of arrival and determines the range based on the ranging protocol. The initialization part of the code configures the timer1 for asynchronous, real time keeping.

It also enables the necessary interrupts and it is here that error correction mechanisms are implemented.

The initialization part of the code forms the repetitive loop of the listener program. When a signal is detected, an interrupt is generated and the execution of the program now goes to the Interrupt Service Routine (ISR), to service the interrupt generated.

The infrared signal is detected at the interrupt-on-change pin. As soon as a valid rising edge appears on this pin, the point of execution in the repetitive loop is recorded and the working and status register values are saved. The interrupt service routine is then invoked. Here, timer1 values are read and recorded as the times of arrival of the infrared signal. A mechanism exists to check that if the low byte of timer1 is read between its rollovers, timer1 is read again to ensure a correct time value. When timer1 has been read, the microcontroller sets a bit that will be used to check for erroneous computations. This bit is referred to as *free* and it is in the *error_check* file register. It then lights a diagnostic LED to show that the arrival of infrared has successfully been reported. The working and Status register values are then restored and the program counter resumes at the point where the interrupt occurred.

The ultrasound signal is detected at the external interrupt pin of the microcontroller. As soon as there is a valid rising edge at the external interrupt pin, the working and status register values are saved and the program counter jumps to the start point of the interrupt service routine. Timer1 values are read, checked for rollover of TMR1L, and recorded as times of arrival of the ultrasound signal.

The ultrasound signal used is 1500 μs long, while the infrared signal is 750 μs long. This means that there can be multiple ultrasound interrupts just before another infrared of another valid sequence of signals is noted. This is where the *free* bit that was set after reading the time of arrival of the infrared signal comes in handy. After the timer1 values are read at the arrival of the ultrasound signal, the *free* bit is checked. If it is found set, the program goes ahead with distance computations.

Distance computations involve finding the time difference of arrival (TDOA) and multiplying it by the speed of sound. It also involves reading the value of the temperature sensor and carrying out the necessary compensations. After the distance values have been sent to the host PC, an ultrasound diagnostic LED is lit. Then, the normal procedure to exit the interrupt service routine is executed.

If the *free* bit is found clear, the distance computation routine is skipped. The ultrasound diagnostic LED is lit and the interrupt service routine is exited. The checking of the *free* bit ensures that no erroneous distance computations are returned. Just by looking at the diagnostic LED's, we cannot tell whether a valid sequence of IR and US signals has been detected. However, the error checking mechanism ensures that only those ultrasound signals that come after infrared signals contribute to distance computations. In other words, distances are computed when and only when an infrared signal followed by an ultrasound signal event is detected at the listener.

Appendix B shows the source code of the programs used in the beacon and listener.

Chapter 5: System Characterization

This chapter describes the experimental setup and gives results of the experimental system characterization as well as the analysis of the performance results. It also describes the challenges encountered and suggests possible solutions to these challenges.

5.1 Experimental Setup

For characterization, two modules were used. The beacon and listener were placed as shown in figure 11. The angle θ was varied and then distance was computed. The distances computed were recorded at the listener over a period of two minutes. With the beacon sending signals every second, this amounts to a total of 120 samples of distances, from which errors were computed. This amount of samples is a best case scenario where all signals sent contribute to distance computation. However, as will be seen in the challenges section, it is not all signals from the beacon that contribute to distance computations. The measured distances were then used to obtain the percentage errors.

5.2 Performance Results

This section shows graphs that were obtained from the various system performances. The graphs show percentage errors resulting from the measured distances under different conditions.

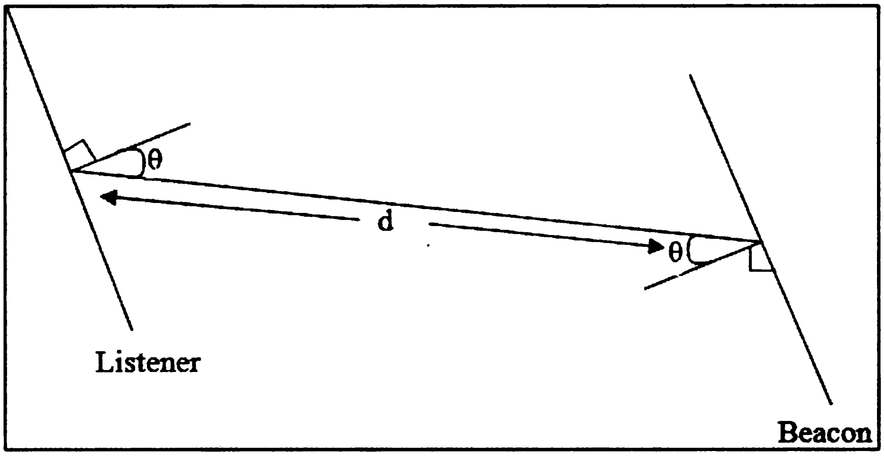


Figure 11: Experimental Setup [1]

5.2.1 Effect of distance and angle of rotation

Figure 13 shows the percentage error in computed distance with increase in distance and change in the angle of rotation. The graph shows that error in distance computation is minimized when the beacon and listener are directly opposite each other (angle of rotation=0 degrees). As the angle of rotation increases, the ultrasound signal strength at the listener decreases, and it takes the listener a long time to detect it, thus contributing to the error increase. This property can be attributed to the nature of ultrasound transducers used in the system. As can be seen in figure in figure 15, the ultrasound signal transmitted can only be detected at a limited range of angles of rotation.

As the nodes move away from each other, the ultrasound signal is attenuated, and it takes the listener a long time to detect it, thus contributing to the error increase in computed distances [1].

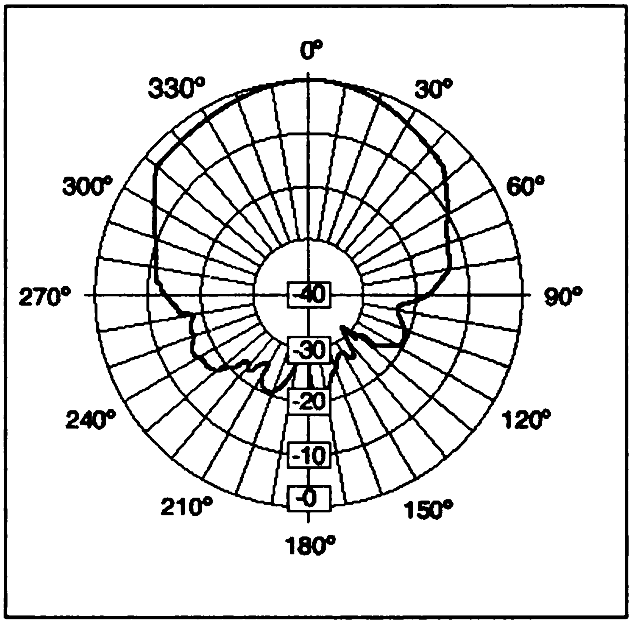


Figure 12: Directivity of US transducers used in the system; directly taken from the datasheet [30]

5.2.2 Effect of ambient light on computed distance

The beacon and listener were placed directly opposite each other (0 degrees) and the distance separating them was varied. Three such experiments were performed under different ambient light conditions: a lit lab during the day, a dark lab, and outdoor. The graph shown in figure 14 below was then plotted from the results.

Outdoors, the system performed poorly and the results of the experiments were not consistent. This poor performance can be attributed to more interference in outdoor environments such as other sources of ultrasound. When we compare the errors observed in the lit and dark labs, we can conclude that ambient light has negligible effect on system performance. The infrared transceivers used are properly shielded from other light source interferences, hence the consistent performance in different ambient light conditions.

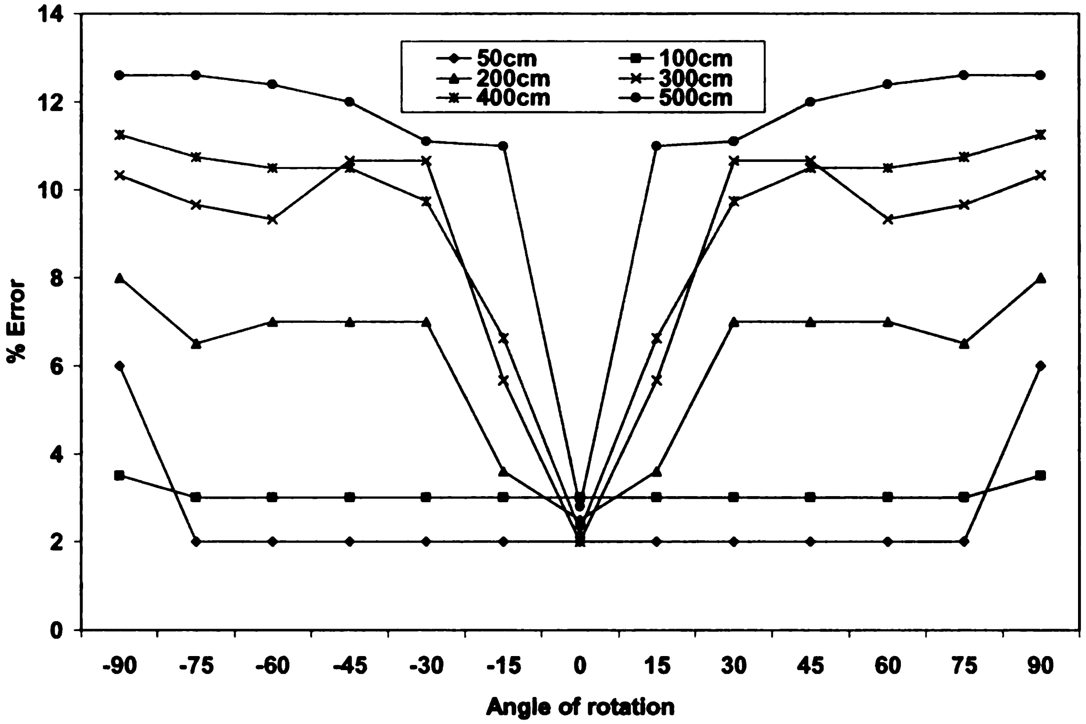


Figure 13: Percentage error in computed distance with increase in distance and angle of rotation

5.2.3 Effect of temperature on system accuracy

The speed of sound changes by approximately 60cm/s for every one degree Celsius increment in ambient temperature [29]. This property can introduce errors in distance computations. For this reason, a temperature sensor was incorporated in the system to help the system adjust the speed of sound according to the ambient temperature.

To show the effect that temperature change can have on the system accuracy, distances were computed at various temperatures, first without the temperature sensor (with a fixed speed of sound=344m/s), and then with the temperature sensor. The graph shown in figure 15 was then plotted. The graph shows that compensations with temperature change help to reduce errors in computed distance, by an average of approximately 7cm.

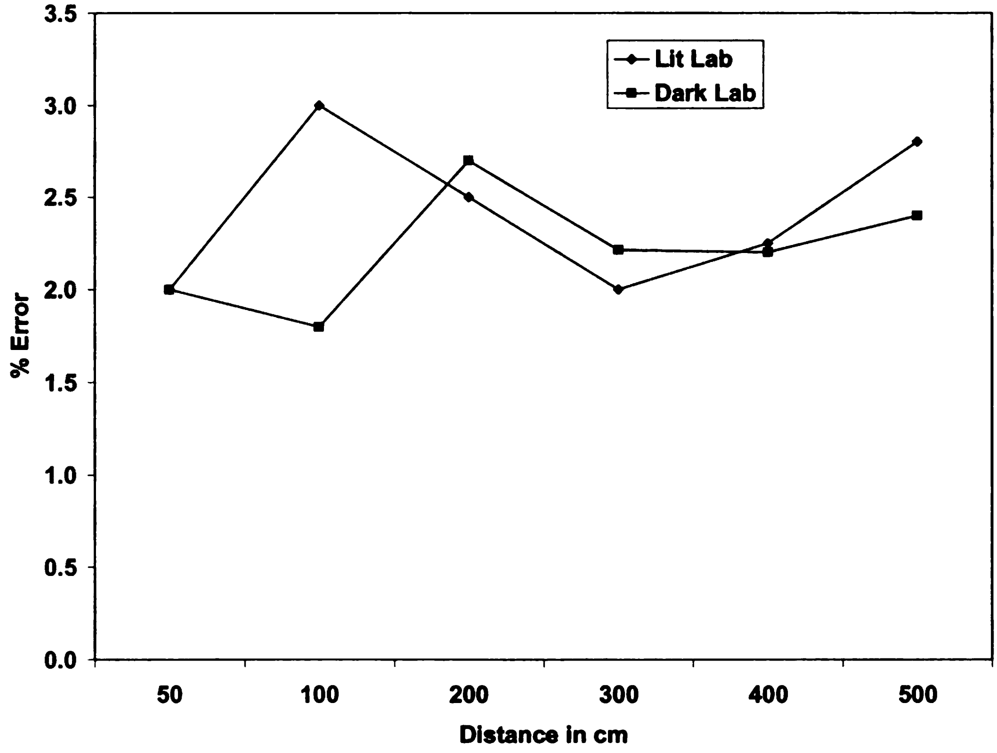


Figure 14: Error in computed distance in different light conditions

5.2.4 Comparisons of errors at short distances between the Cricket and the IRUS

The IRUS indoor location system can be adjusted for short distance computations by adjusting the ultrasound signal strength at the beacon and listener. After this adjustment was made, the IRUS accuracy with measured short distances was compared with that of the Cricket system. This comparison is shown in figure 16.

The graph shows that for most short distances measured, the IRUS system has a percentage error less than 10.

5.2.5 Comparisons of errors at long distances between the cricket and the IRUS

The IRUS system was adjusted for long distance computation and the resulting errors were compared to those claimed by the cricket team. The resulting graph (figure 17) shows an error difference of about 1.2% between the two systems. The ultrasound signal is greatly attenuated beyond 8m and therefore no distances were computed beyond this value.

From the two preceding graphs (figures 16 & 17), we conclude that the IRUS system performs within the expected ranges of an indoor location system such as the MIT Cricket.

5.2.6 Error in computed distance observed at the oscilloscope

With the beacon and listener directly opposite each other, signals sent were observed at the oscilloscope, as shown in figure 18. The time difference of arrival was obtained and multiplied by 344 (speed of sound at room temperature).

The resulting errors in computations of various distances were then plotted against the measured distances, as shown in figure 19. The graph shows that at long distances, there was an error increase. The distance computations were affected by the scope's resolution, hence the observed increase in error in computed distances. Also, by looking at the signals arriving at the listener, it is impossible to tell exactly what edge causes an interrupt, hence the possibility of errors. The results from this experiment also demonstrate that the routine running on the listener can be trusted.

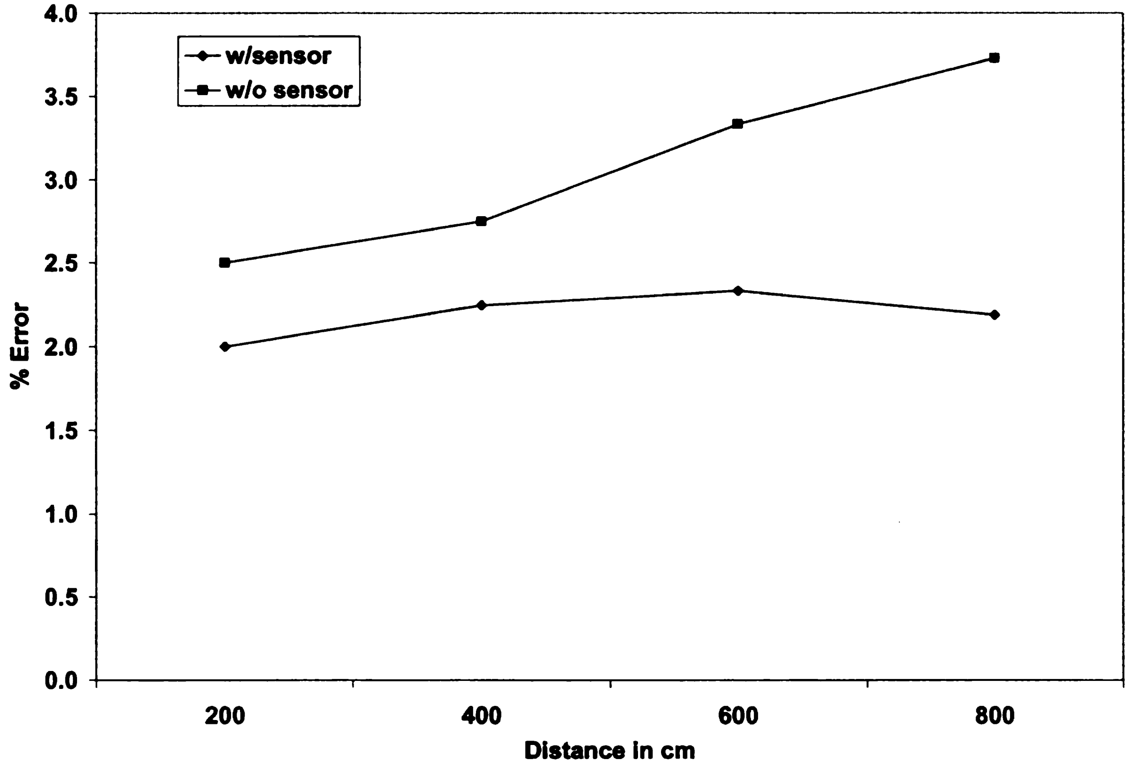


Figure 15: Error with distance increase shown with and without temperature compensation

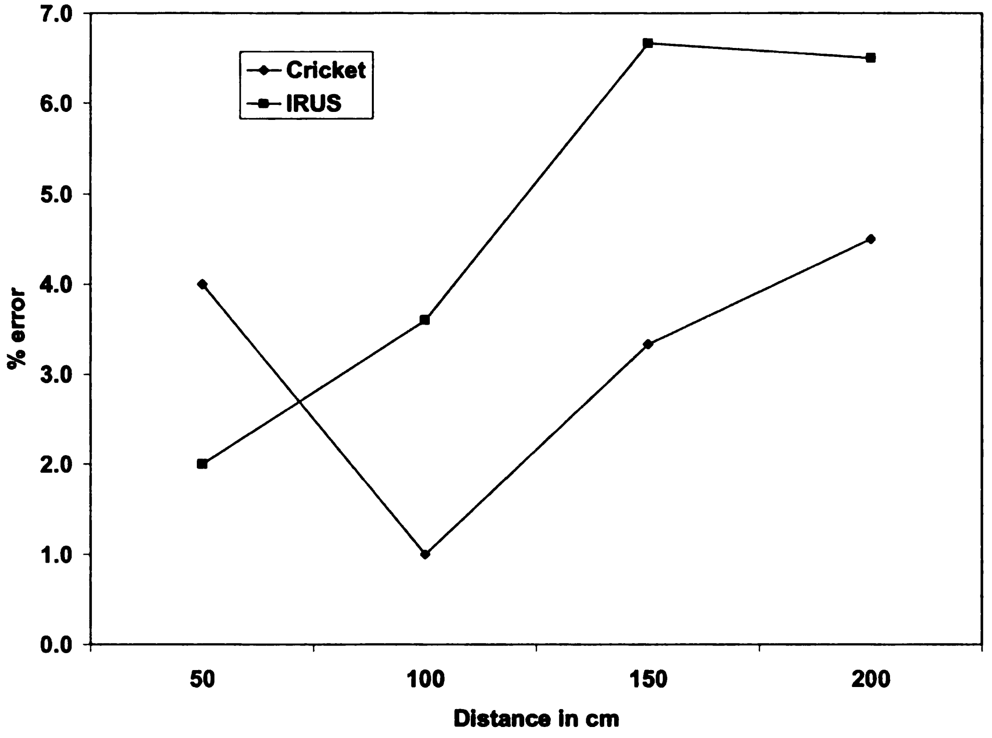


Figure 16: Error with computed distance at short distances

5.3 Challenges

Figure 20 is a representation of both signals as they leave the beacon. The figure shows a delay before the production of the ultrasound signal. This delay is negligible and therefore it is assumed that the signals leave the beacon at the same time.

Figure 21 shows the structure of the infrared signal that is sent from the beacon. The signal is a narrow beam, which is IrDA compliant. Figure 22 and 23show a zoomed in structure of the infrared signal.

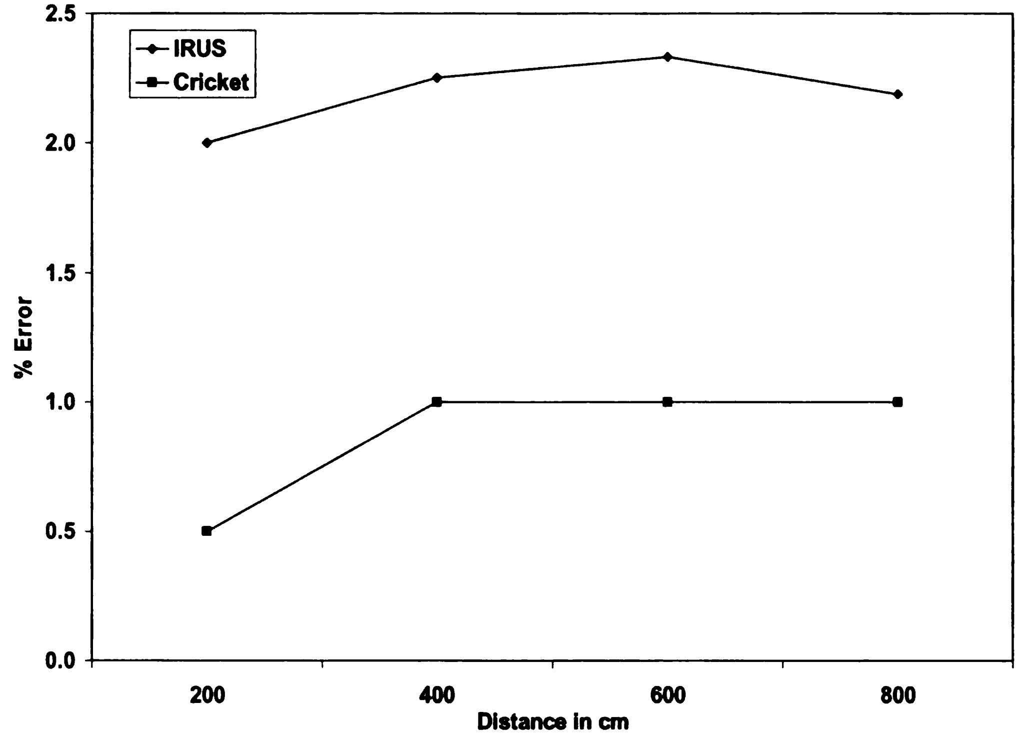


Figure 17: Error in computed distance at long distances

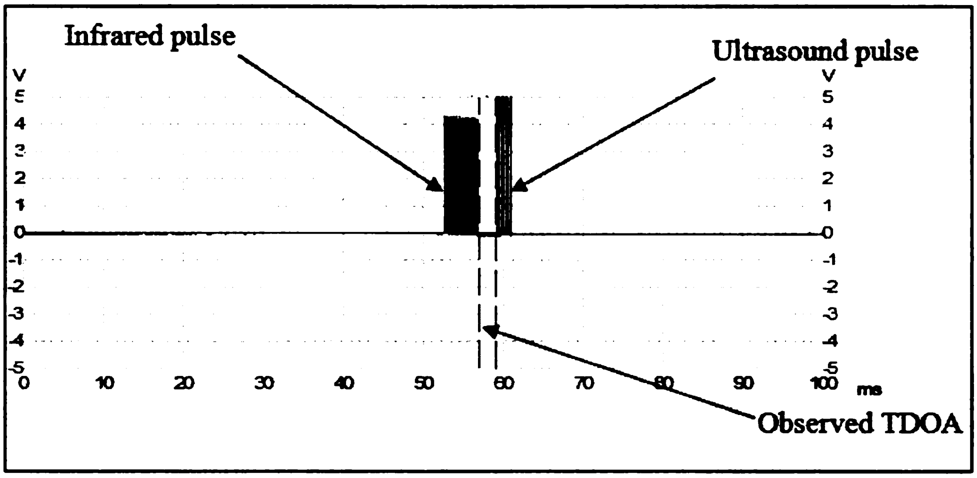


Figure 18: TDOA determination at the scope

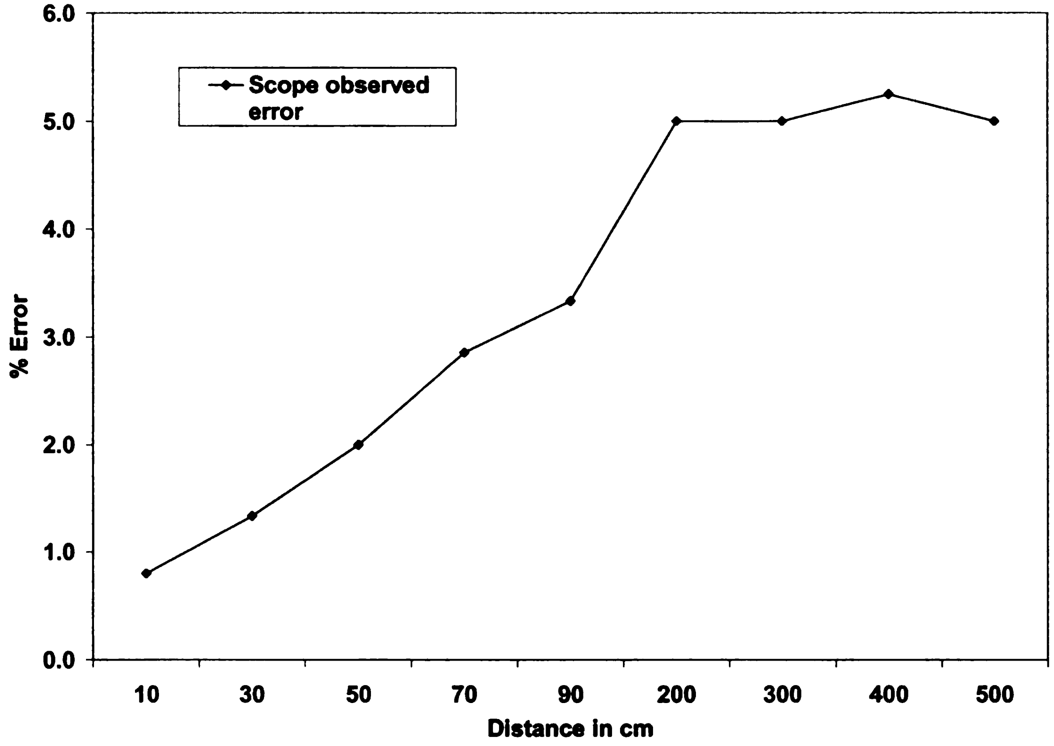


Figure 19: Error in computed distance observed at the oscilloscope

The arrival of an infrared signal is detected when there is a rising edge at the interrupt on-change pin of the microcontroller. Since there is a possibility that two arrival events of infrared happen consecutively, before the arrival of an ultrasound signal, it is likely to have two values for the same distance estimate. However, such occurrences are very rare and would be very likely in cases involving long distances where ultrasound signals delay. In case they happen, the difference in the values computed is not significant since the infrared signal is very narrow.

Figure 24 shows a sequence of US signals as they leave the beacon. Figure 25 is a zoomed in representation of the US signal. Note that the pulse length is approximately 1500 μs. It is slightly greater than 1500 due to errors introduced by delay routines and clock precision.

The arrival of an ultrasound signal at the listener is detected when there is a valid rising edge at the external interrupt pin of the microcontroller. To calculate the distance between the beacon and the listener, the listener has to detect both the infrared and ultrasound signals within an allowable time frame. False distance computations due to multiple arrivals of US signals before IR signals arrive are avoided in the algorithm running on the listener. Here, it is ensured that no distance computation is done before the IR arrival event followed by the US arrival event happen.

Signals at the beacon are generated through delay routines. Delay routines in themselves have errors, which can be reflected in the values of distances computed. For instance, figure 24 shows a recurring error caused by software delay routines at the beacon. This error can be corrected by filtering out values corresponding to this TDOA.

As was described in the software architecture, the listener runs a repetitive loop until a signal arrives, after which an interrupt is generated. There are several interrupts competing for system resources and this feature can lead to erroneous distance computations.

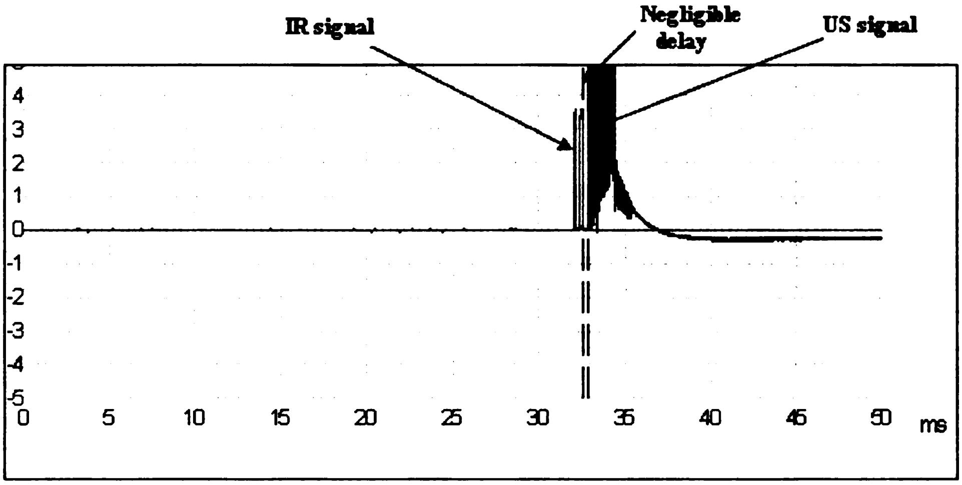


Figure 20: Beacon generated signals

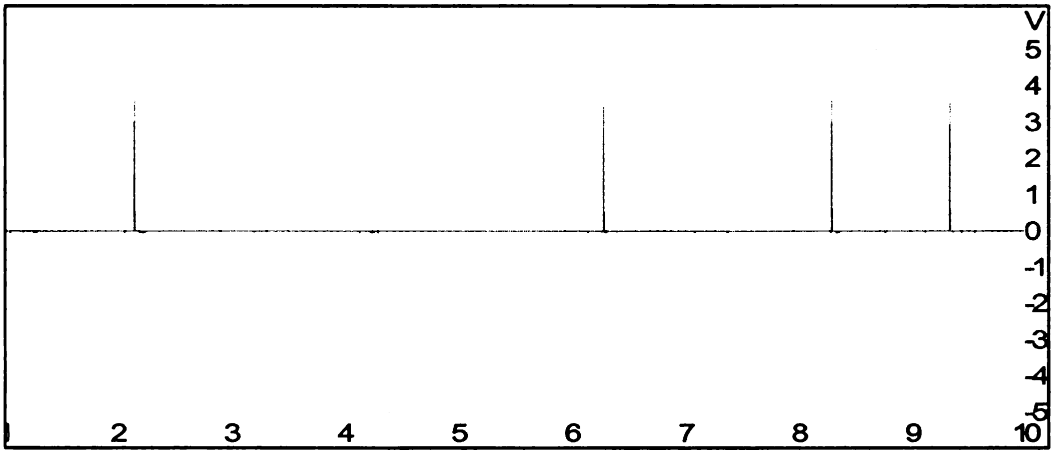


Figure 21: Infrared signal structure

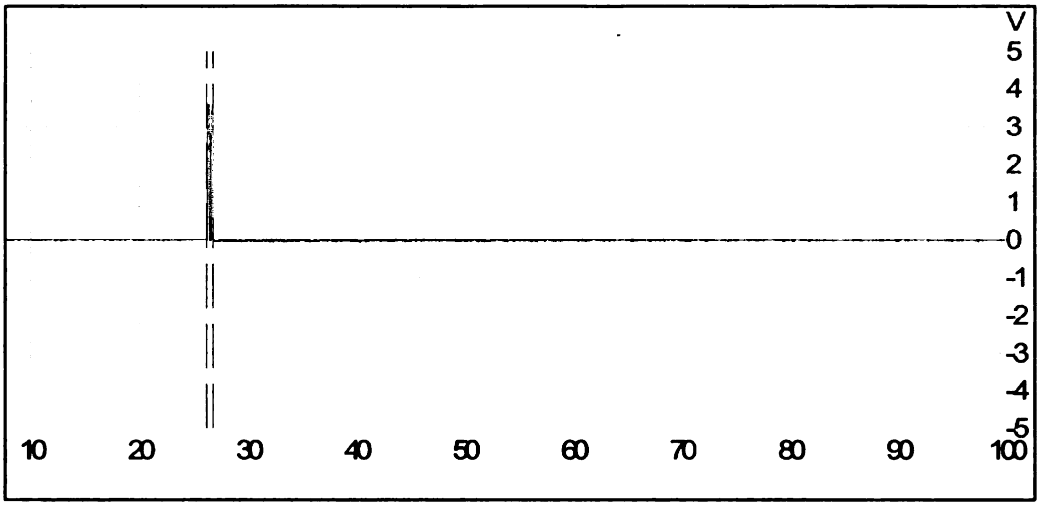


Figure 22: Zoomed-in structure of the infrared signal

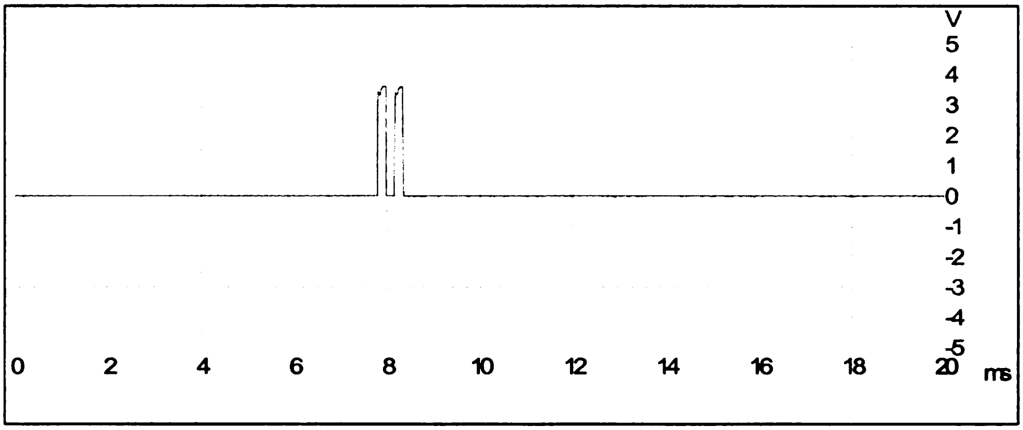


Figure 23: Zoomed-in structure of an infrared signal (2)

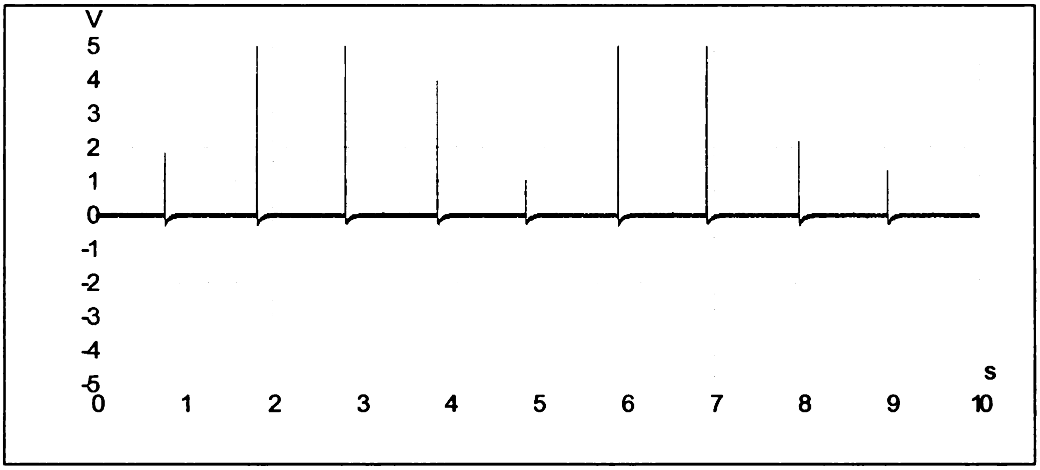
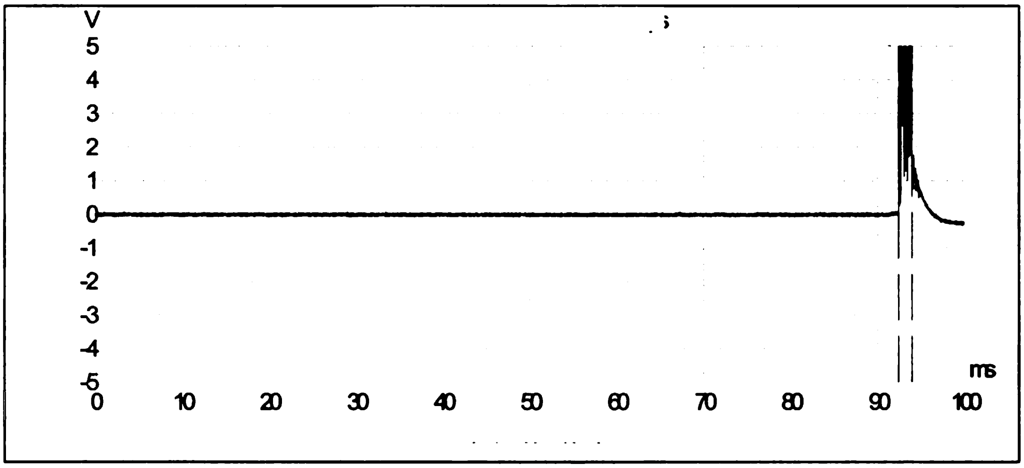
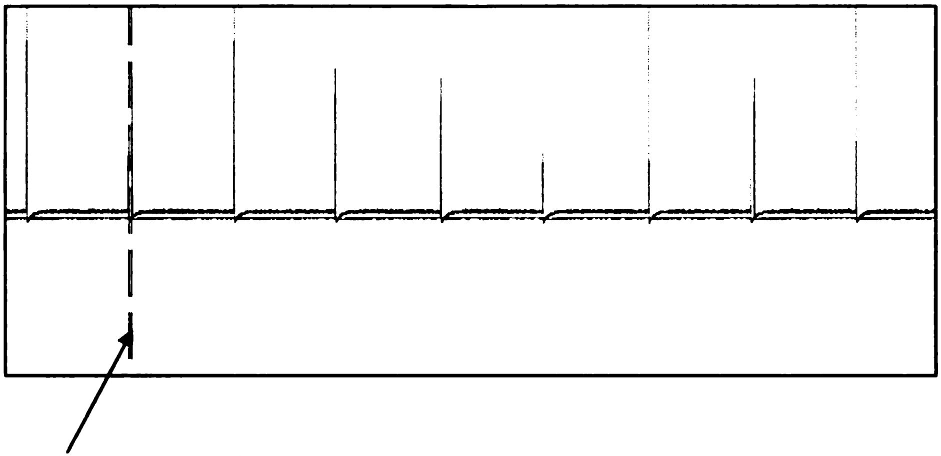


Figure 24: Ultrasound signal structure



25: Zoomed-in structure of an US signal



Outlier TDOA value

Figure 26: Recursive Error caused by delay routines at the beacon

Chapter 6: **Conclusions**

In this thesis, the design, implementation, and characterization of a new indoor location system has been presented. It has been shown that the time difference of arrival (TDOA) algorithm, developed by the MIT Cricket team; can be applied to a new combination of signals: Infrared and Ultrasound. This new combination ensures that interferences between nodes in neighboring rooms is avoided since both ultrasound and infrared signals do not penetrate walls.

A novel system of indoor localization has been designed. This system is based on range computation using the time difference of arrival of infrared signals and ultrasound signals. The system architecture has been presented, followed by its characterization in different ambient conditions and experimental results.

It has been shown that a location system based on TDOA of infrared and ultrasound signals can be implemented in readily available off-the-shelf components. Possible advantages of such a system have been explored and presented.

The system developed and presented in this thesis can be modified to compute short distances e.g. from 5-200 cm. At the same time, it can operate at longer distances: from 2-8 m. Characterization results have shown that the system's performance at short as well as long distances is commendable.

Chapter 7: **Future Work**

The system presented in this thesis work is intended for experimentation. In future, multiple units will be assembled and other aspects of localization will be investigated. For instance, by building two more beacons, we will enable the listener to infer its position from the messages sent by the three beacons. This feature will call for the inclusion of a message send and receive mechanism. The listener should therefore be able to tell which beacon sent what message.

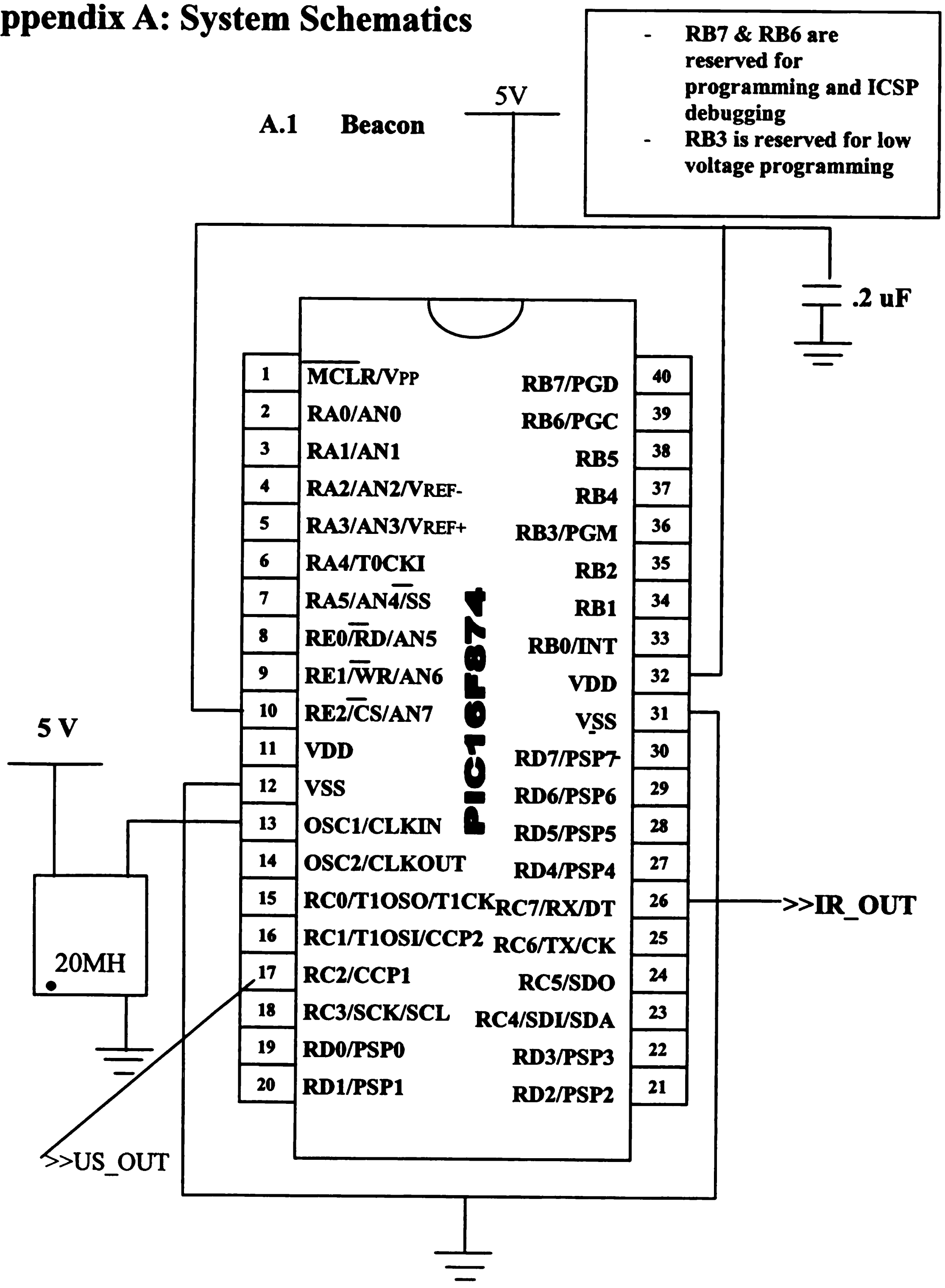
The use of multiple modules will call for miniaturization. The components used in the system architecture are available in surface mount technology, and therefore the system can be miniaturized and industrially produced.

For commercialization, both functions of beacon and listener will be combined in one module and the option of configuring the module as either beacon or listener will be left to the user.

Once the characterization of multiple modules has been done, applications will be developed based on the system. Some of the applications that can be experimented with are asset tracking and robot navigation.

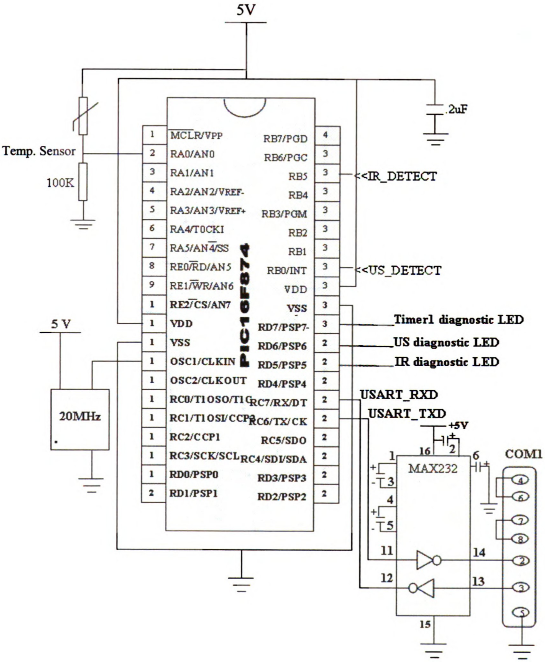
Appendix A: System Schematics

A.1 Beacon



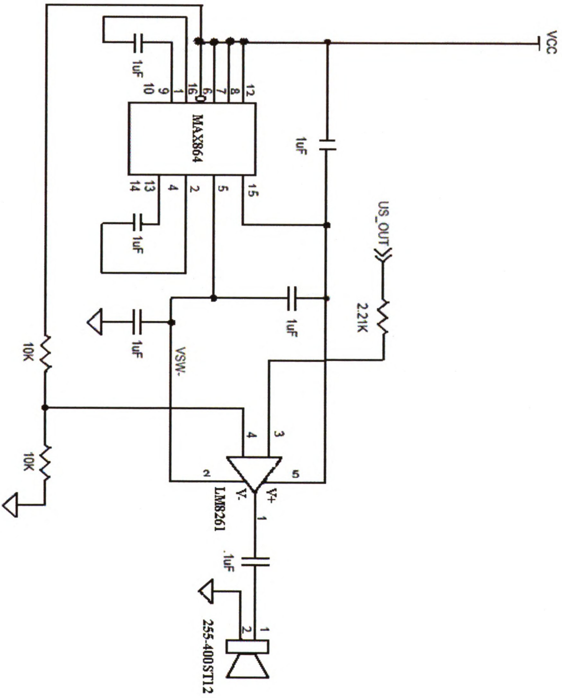
- RB7 & RB6 are reserved for programming and ICSP debugging
- RB3 is reserved for low voltage programming

A.2 Listener

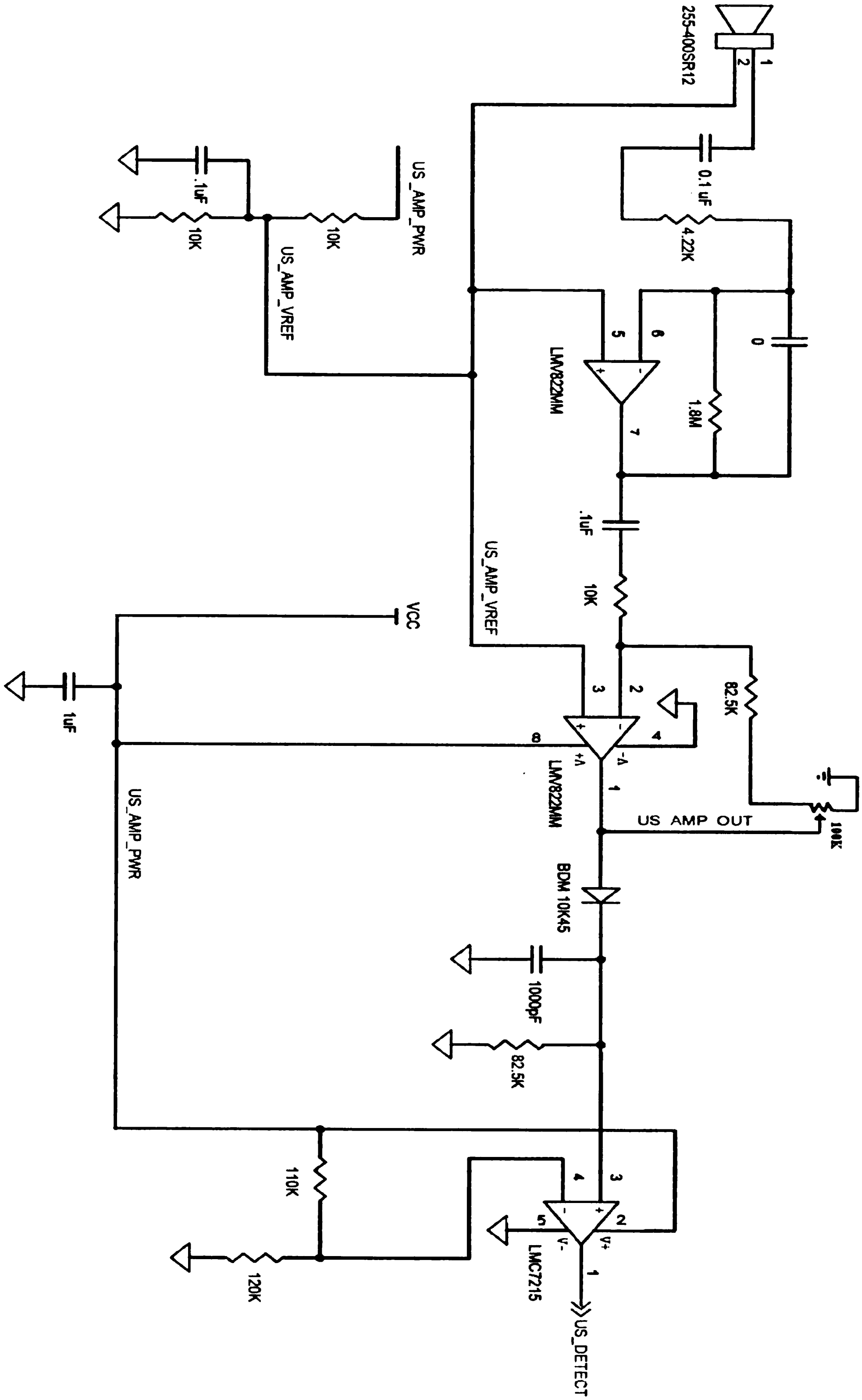

5V
.2uF
Temp. Sensor
100K
MCLR/VPP
RA0/AN0
RA1/AN1
RA2/AN2/VREF-
RA3/AN3/VREF+
RA4/T0CKI
RA5/AN4/SS
RE0/RD/AN5
RE1/WR/AN6
RE2/CS/AN7
VDD
VSS
OSC1/CLKIN
OSC2/CLKOUT
RC0/T1OSO/T1C
RC1/T1OSI/CCP2
RC2/CCP1
RC3/SCK/SCL
RD0/PSP0
RD1/PSP1
RB7/PGD
RB6/PGC
RB5
RB4
RB3/PGM
RB2
RB1
RB0/INT
VDD
VSS
RD7/PSP7
RD6/PSP6
RD5/PSP5
RD4/PSP4
RC7/RX/DT
RC6/TX/CK
RC5/SDO
RC4/SDI/SDA
RD3/PSP3
RD2/PSP2
PIC16F874
<<IR_DETECT
<<US_DETECT
Timer1 diagnostic LED
US diagnostic LED
IR diagnostic LED
USART_RXD
USART_TXD
+5V
5 V
20MHz
MAX232
COM1
1
2
3
4
5
6
11
12
13
14
15
16

A.3 Ultrasound Section [1]

A.3.1 Transmitter



A.3.2 Receiver



Appendix B: System Assembly Program

B.1 Beacon

```
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~
;This program has parts taken from application notes provided by Microchip company

;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~
;
;       Filename: Beacon
;       Date:   Aug 17, 2007
;       File Version:  1.0
;       Assembled using: MPLAB IDE 7.60.00.0
;
;       Author:         David Rurangirwa
;       Company:NEEWS LAB of MSU
;
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~
;
;  Files required:              p16f874.inc
;
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~
;
;       Program Description:
;
;This program enables the PIC16F874 to periodically send a signal
;through the SPI protocol which is converted into an IR signal by other
;peripherals; and Ultrasound signal which is generated through the PWM
;mode of the PIC. The signals are sent with a delay of approx. 9.4 us between
;them and this procedure is repeated every 1 s. The delay routines in this
;program were taken from the delay code generator at
;http://www.piclist.com/techref/piclist/codegen/delay.htm
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~

 LIST P=16F874

; 20MHz crystal is being used, thus each instruction cycle = 200nS
```

```
;*******************RAM register definitions*********************

outbyte1            equ 0x20
outbyte2            equ 0x21
temp1               equ 0x22
temp2               equ 0x23
nmsec0              equ 0x29
nmsec1              equ 0x24
nmsec2              equ 0x25
nmsec_off0          equ 0x26
nmsec_off1          equ 0x27
nmsec_off2          equ 0x28

;*******************Bit definitions******************************
CS                  equ 7

;*******************Include file**********************************
 include "p16f874.inc"              ;include file for a PIC16F877
 errorlevel  -302                ; suppress message 302 from list file

;*****************************************************************
 org 0x000                 ; set the reset vector
 goto main                ; go to the beginning of program

;!!!!!!!!!!!!!!!!!!!!!!Begin Main Program!!!!!!!!!!!!!!!!!!!!!!!!

main

IR_INI
              bcf       STATUS,RP0          ; set to bank 0
              clrf      PORTC             ; initialize portc to 0
              bsf       STATUS,RP0          ; set to bank 1
              movlw     0x10             ; all bits are outputs except SDI
              movwf     TRISC              ; move the value to TRIS portc
              bcf       STATUS,RP0          ; set to bank0
              bsf       PORTC,CS            ; make the chip select high
              clrf      PIE1             ; disable peripheral interrupts
              clrf      INTCON             ; disables all interrupts
              bcf       STATUS,RP0          ; set to bank 0
              clrf      SSPCON             ; clear SSP control register
              movlw     0x20             ; set up spi port, SPI master,
              movwf     SSPCON              ; clk/16, ckp=1 (mode 1,1)
```

```
bsf             STATUS,RP0          ; set to bank 1
                clrf        SSPSTAT             ; clear SSP status register
                movlw       0x80            ; set up spi port, SPI master,
                movwf       SSPSTAT             ; cke = 0 (mode 1,1)

        ;Send the read enable sequence (RDSR)
                bcf         STATUS,RP0                  ;bank1
                bcf         PORTC,CS            ; clear the chip select line
                movlw       0x40            ; load RDSR sequence
                movwf       outbyte1        ; store in RAM location outbyte
                movlw       0x01            ;load teh LSB of RDSR
                movwf       outbyte2        ;store in RAM location outbyte
                call        output          ;send the sequence
                bsf         PORTC,CS        ; set the chip select line



Loop    ;this part of the program repeats
                call        IR_Data         ;send infrared pulse
                call        delay_10cm      ;compensate for constant error of 10cm
                call        US_Data         ;send Ultrasonic pulse
                call        Delay_off       ;wait for 1 s
                goto        Loop            ;before sending again

IR_Data

                ;Send the write enable sequence (WREN)

                bcf         PORTC,CS        ; clear the chip select (active)
                movlw       0xC0             ; load WREN sequence
                movwf       outbyte1         ; store in RAM location outbyte
                movlw       0x0A            ;baud=9600bps
                movwf       outbyte2        ;store in RAM location outbyte
                call        output          ;send the sequence
                bsf         PORTC,CS            ; set the chip select line
                bcf         PORTC,CS            ; clear the chip select line
                movlw       0x80            ; load WRITE sequence
                movwf       outbyte1            ; store in RAM location outbyte
                movlw       0xAA        ; load high address byte
                movwf       outbyte2            ; store in RAM location outbyte
                call        output          ; call the SPI output routine
                bsf         PORTC,CS            ; set the chip select line
               return          ;go to position where routine was last called
```

```
output
            bcf       STATUS,RP0      ;bank0
            movf      outbyte1,w       ; move outbyte1 into w
            movwf     SSPBUF           ; place data in send buffer
            call      delay            ;allow some time lapse
            movf      outbyte2,w       ;move outbyte2 into w
            movwf     SSPBUF            ;place data in send buffer
            call      delay              ;allow some time lapse
            return                  ;go to position where routine was last called

delay
            movlw     0x32          ; move literal value into w
            movwf     temp1          ; move w value into temp1


            movlw     0xFA           ; move literal value into w
            movwf     temp2          ; move w value into temp2


            decfsz    temp2,f       ; decrement temp2, skip if zero
            goto      $-1           ; goto decrement 2 if not zero
            decfsz    temp1,f        ; decrement temp1, skip if zero
            goto       $-1          ; goto decrement 1 if not zero
            return                  ;return both locations = 0


US_Data

           ;routine to initialize production of US signal
            bsf       STATUS,RP0         ;move to bank 1
            movlw     0x7C               ;set the PWM period by writing to
PR2
            movwf     PR2
            bcf       STATUS,RP0         ;Bank0
            movlw       0x3E             ;duty cycle=50%
            movwf     CCPR1L ;           ;set by writing to CPR1L
            call      Delay_on           ;signal is on for 150us
            movlw       0x00
            movwf       CCPR1L
```

```
;initiazile the timer and set the prescale of TMR2
                bcf         STATUS,RP0          ;bank0
                clrf        TMR2                ;clear the TMR2 register
                clrf        T2CON               ;clear the Timer2 control register
                movlw       0x2C                ;Configure the CCP1 module for
PWM operation
                movwf       CCP1CON             ;by writing into the CCP control
register
                movlw        0x04               ;turn on the Timer2, prescale is 1:1
                movwf        T2CON
                return


Delay_on
                movlw       0x26        ;delay values for 1500us
                movwf       nmsec1
                movlw       0x24
                movwf       nmsec2

Delay_yus
                decfsz      nmsec1,F
                goto        $+2
                decfsz      nmsec2,F
                goto        Delay_yus
                goto        $+1
                nop
                return


Delay_off                               ;delay of 1 s
                movlw       0x2C
                movwf       nmsec0
                movlw       0xE7
               movwf        nmsec1
                movlw       0x0B
                movwf       nmsec2
Delay_xms
                decfsz      nmsec0,F
                goto        $+2
                decfsz      nmsec1,F
                goto         $+2
                decfsz      nmsec2,F
```

```
 goto            Delay_xms
                  goto          $+1
                  return
delay_10cm
                  movlw         0x2A
                  movwf         nmsec0
                  movlw         0x02
                  movwf         nmsec1
Delay_0
                  decfsz        nmsec0, f
                  goto          $+2
                  decfsz        nmsec1,f
                  goto          Delay_0
                  goto          $+1
                  return


                  END
```

B.2 Listener

```
;-------------------------------------------------------------------
;
;  Filename:      Listener.asm
;  Date:          July 30th, 2007
;  File Version:  4.0
;
;  Author:        David Rurangirwa
;  Company:       NEEWS Lab of MSU
;
;
;-------------------------------------------------------------------
;
;  Files required:
;  1.  Fxd26.a16
;  2.  FXM66.a16
;  3.  FP32.a16
;
;-------------------------------------------------------------------
```

```
;
;Notes:  This program enables the listener microcontroller to
;        detect infrared and ultrasound signals sent from the beacon,
;        record their times of arrival and find the difference in the times
;        of arrival, then calculates the distance between the listener and beacon.
;        It features a compensation mechanism where the speed of sound is adjusted
;        according to changes in ambient temperature.
;-----------------------------------------------------------------


        list     p=16f874            ; list directive to define processor
        #include <p16f874.inc>       ; processor specific variable definitions
        #include <math16.inc>
        errorlevel -302           ;suppress "not in bank 0" message

        __CONFIG  _CP_OFF  &  _WDT_ON  &  _BODEN_ON  &  _PWRTE_ON  &
_RC_OSC & _WRT_ENABLE_ON & _LVP_ON & _CPD_OFF

; '__CONFIG' directive is used to embed configuration data within .asm file.
; The lables following the directive are located in the respective .inc file.
; See respective data sheet for additional information on configuration word.
;----------------------------------------------------------------------

;***** Constants
SPBRG_VAL EQU   .129              ;set baud rate 9600 for 20Mhz clock
SIG_FIG equ 8 ;set SIG_FIG equal to the number of significant figures in your decimal
number
;for example: ones, tenths,hundredths,thousandths, requires 4 sig figs
last_digit      set ones
flag            equ   10
TEMP_ad         equ   11
adover          equ   0
adif            equ   1
adgo            equ   2
adie            equ   6
gie             equ   7
rp0             equ   5
free            equ           3
;----------------------------------------------------------------------
```

```
;Variables

            CBLOCK      0x70
            WREG_TEMP               ;storage for WREG during interrupt
            STATUS_TEMP             ;storage for STATUS during interrupt
            PCLATH_TEMP             ;storage for PCLATH during interrupt
            FSR_TEMP        ;storage for FSR during interrupt
            temperature     ;storage for ambient temperature
            error_check
            ROC                     ;storage for rollover count
            d1
            d2
            d3
            ENDC

            CBLOCK      0x50
            ACCbROC
            ACCbHI                  ;Vus, MSB and part of d
            ACCbLO                  ;Vus, LSB and part of d
            ACCcHI                  ;part of d
            ACCcLO                          ;part of
            EXPb                    ;Vus, EXP and exp of d
            ACCdHI
            ACCdLO
            ACCeHI
            ACCeLO
            ENDC

            CBLOCK      0x40
            TMPROC_IR
            TMPROC_US
            TMPH_IR                         ;IR arrival time, MSB
            TMPL_IR                         ;IR arrival time, LSB
            TMPH_US                         ;US arrival time, MSB
            TMPL_US               ;US arrival time, LSB
            ACCaROC                         ;TDOA,MSB
            ACCaHI                ;TDOA, MSB
            ACCaLO                ;TDOA, LSB
            ROCH_IR
            ROCH_US
            ROCL_IR
            ROCL_US
```

```
vusl                    ;low byte of vus
        ENDC

        CBLOCK  0x60
        tmillions
        millions
        hundredthousands
        tenthousands
        thousands
        hundreds
        tens
        ones
        temp
        digit_count ;counter used to cycle through each digit
        ENDC


;---------------------------------------------------------------------------
;Macros to select the register bank
; Many bank changes can be optimized when only one STATUS bit changes

Bank0   MACRO                   ;macro to select data RAM bank 0
        bcf    STATUS,RP0
        bcf    STATUS,RP1
               ENDM

Bank1   MACRO                   ;macro to select data RAM bank 1
        bsf    STATUS,RP0
        bcf    STATUS,RP1
               ENDM

Bank2   MACRO                   ;macro to select data RAM bank 2
        bcf    STATUS,RP0
        bsf    STATUS,RP1
               ENDM

Bank3   MACRO                   ;macro to select data RAM bank 3
        bsf    STATUS,RP0
        bsf    STATUS,RP1
               ENDM
```

```
;---------------------------------------------------------------------------
            ORG    0x000          ; processor reset vector
            clrf    PCLATH          ; ensure page bits are cleared
            goto     main           ; go to beginning of program
;---------------------------------------------------------------------------
; isr code can go here or be located as a call subroutine elsewhere
ISR
            ORG           0x004          ;place code at interrupt vector
            movwf         WREG_TEMP         ;save WREG
            movf          STATUS,W   ;store STATUS in WREG
            clrf          STATUS            ;select file register bank0
            movwf         STATUS_TEMP       ;save STATUS value
            movf          PCLATH,W   ;store PCLATH in WREG
            movwf         PCLATH_TEMP       ;save PCLATH value
            clrf          PCLATH            ;select program memory page0
            movf          FSR,W         ;store FSR in WREG
            movwf         FSR_TEMP   ;save FSR value

;---------------------------------------------------------------------------
INTR_POLL
            Bank0
            btfsc         PIR1, TMR1IF;No timer1 overflow interrupt?
            goto          T1_OVRFL
            BTFSC         INTCON,INTF ; No External interrupt?
            goto          US_INT ;Service interrupt
            BTFSC         INTCON,RBIF;No interrupt on change?
            goto          IR_INT
;---------------------------------------------------------------------------
IR_INT

            Bank0
            MOVF          PORTB,1
            call          RDTMR1_IR

IR_rdend
            bsf           error_check,free  ;set  bit  to  avoid  erraneous  distance
computations
            Bank1
            bcf           TRISD,6;light LED
            Bank0
            BCF           INTCON,RBIF
            goto          EXIT_INT
```

```
US_INT
		Bank0
		call		RDTMR1_US
US_rdend
		btfsc		error_check,free ;check the free bit to determine if IR
signal was received
		call		TDOA ;if so, then calculate distance based on TDOA,
otherwise light LED and exit ISR

comp_done
		Bank1
		bcf		TRISD,7;light LED
		Bank0
		BCF		INTCON,INTF;clear flag
		goto		EXIT_INT

T1_OVRFL
		Bank0
		BCF		PIR1, TMR1IF ; Clear Timer1 Interrupt Flag
		Bank1
		bcf		TRISD,5
		goto		EXIT_INT
;-----------------------------------------------------------------------------
;Time recordings of IR and US signals

RDTMR1_IR
		Bank0
		CLRF		INTCON		;All interrupts are disabled
		MOVF		TMR1H, W ; Read high byte
		MOVWF		TMPH_IR ;store in temporary register
		MOVF		TMR1L, W ; Read low byte
		MOVWF		TMPL_IR ;store in temporary register
		MOVF		TMR1H, W ; Read high byte
		SUBWF		TMPH_IR, W ; Sub 1st read with 2nd read
		BTFSC		STATUS,Z ; Is result = 0
		return;		Good 16-bit read
;
; TMR1L may have rolled over between the read of the high and low bytes.
; Reading the high and low bytes now will read a good value.
```

```
;
            MOVF        TMR1H, W ; Read high byte
            MOVWF       TMPH_IR ;
            MOVF        TMR1L, W ; Read low byte
            MOVWF       TMPL_IR ;
; Re-enable the Interrupt (if required)
            goto        IR_rdend
;--------------------------------------------------------------------------
RDTMR1_US
            Bank0
            CLRF        INTCON      ;All interrupts are disabled
            MOVF        TMR1H, W ; Read high byte
            MOVWF       TMPH_US ;store in temporary register
            MOVF        TMR1L, W ; Read low byte
            MOVWF       TMPL_US ;store in temporary register
            MOVF        TMR1H, W ; Read high byte again
            SUBWF       TMPH_US, W ; Sub 1st read with 2nd read
            BTFSC       STATUS,Z ; Is result = 0?
            return

; TMR1L may have rolled over between the read of the high and low bytes.
; Reading the high and low bytes now will read a good value.
;
            MOVF        TMR1H, W ; Read high byte
            MOVWF       TMPH_US ;
            MOVF        TMR1L, W ; Read low byte
            MOVWF       TMPL_US ;
; Re-enable the Interrupt (if required)
            goto        US_rdend
;----------------------------------------------------------------------------
TDOA
            MOVFTMPL_IR,W
            SUBWF       TMPL_US,0                       ;   find    difference
between high byte values
            MOVWF       ACCaLO                              ;           store
difference
            MOVF        TMPH_IR,W
            btfss       STATUS,C    ;add in carry
            incfsz      TMPH_IR,W
            subwf       TMPH_US,0
            MOVWF       ACCaHI                              ;           store
difference
```

```
call            ReadADC                                         ;read temperature sensor
                call            LookupTable                             ;compare  with  table
values to determine Vus
;----------------------------------------------------------------------------

loadAB
                movlw           0x01
                movwf           ACCbHI
                movf            vusl,w
                movwf           ACCbLO      ;; loads  ACCb  =  344m/s,  floating  point
notation of PIC for 344
                goto            F_mpy


;------------------------------------------------------------------------
;Lookup table to vary Vus
LookupTable
                movlw           0x23
                subwf           temperature,w
                btfsc           STATUS,C
                goto            NextLookup22
                movlw           0x58
                movwf           vusl
                goto            loadAB

NextLookup22
                movlw           0x24
                subwf           temperature,w
                btfsc           STATUS,C
                goto            NextLookup23
                movlw           0x58
                movwf           vusl
                goto            loadAB

NextLookup23
                movlw           0x25
                subwf           temperature,w
                btfsc           STATUS,C
                goto            NextLookup24
                movlw           0x59
                movwf           vusl
                goto            loadAB
```

```
NextLookup24
            movlw       0x26
            subwf       temperature,w
            btfsc       STATUS,C
            goto        NextLookup25
            movlw       0x59
            movwf       vusl
            goto        loadAB

NextLookup25
            movlw       0x27
            subwf       temperature,w
            btfsc       STATUS,C
            goto        NextLookup26
            movlw       0x59
            movwf       vusl
            goto        loadAB

NextLookup26
            movlw       0x28
            subwf       temperature,w
            btfsc       STATUS,C
            goto        NextLookup27
            movlw       0x5A
            movwf       vusl
            goto        loadAB

NextLookup27
            movlw       0x29
            subwf       temperature,w
            btfsc       STATUS,C
            goto        NextLookup28
            movlw       0x5A
            movwf       vusl
            goto        loadAB

NextLookup28
            movlw       0x2A
            subwf       temperature,w
            btfsc       STATUS,C
            goto        NextLookup29
            movlw       0x5A
```

```
movwf         vusl
              goto          loadAB

NextLookup29
              movlw         0x2B
              subwf         temperature,w
              btfsc         STATUS,C
              goto          NextLookup30
              movlw         0x5B
              movwf         vusl
              goto          loadAB

NextLookup30
              movlw         0x2C
              subwf         temperature,w
              btfsc         STATUS,C
              goto          NextLookup8
              movlw         0x5B
              movwf         vusl
              goto          loadAB

NextLookup8
              movlw         0x16
              subwf         temperature,w
              btfsc         STATUS,C
              goto          NextLookup9
              movlw         0x54
              movwf         vusl
              goto          loadAB

NextLookup9
              movlw         0x17
              subwf         temperature,w
              btfsc         STATUS,C
              goto          NextLookup10
              movlw         0x54
              movwf         vusl
              goto          loadAB
```

```
NextLookup10
          movlw     0x18
          subwf     temperature,w
          btfsc     STATUS,C
          goto      NextLookup11
          movlw     0x54
          movwf     vusl
          goto      loadAB

NextLookup11
          movlw     0x19
          subwf     temperature,w
          btfsc     STATUS,C
          goto      NextLookup12
          movlw     0x55
          movwf     vusl
          goto      loadAB

NextLookup12
          movlw     0x1A
          subwf     temperature,w
          btfsc     STATUS,C
          goto      NextLookup13
          movlw     0x55
          movwf     vusl
          goto      loadAB

NextLookup13
          movlw     0x1B
          subwf     temperature,w
          btfsc     STATUS,C
          goto      NextLookup14
          movlw     0x55
          movwf     vusl
          goto      loadAB

NextLookup14
          movlw     0x1C
          subwf     temperature,w
          btfsc     STATUS,C
          goto      NextLookup15
          movlw     0x56
          movwf     vusl
          goto      loadAB
```

```
NextLookup15
                movlw       0x1D
                subwf       temperature,w
                btfsc       STATUS,C
                goto        NextLookup16
                movlw       0x56
                movwf       vusl
                goto        loadAB

NextLookup16
                movlw       0x1E
                subwf       temperature,w
                btfsc       STATUS,C
                goto        NextLookup17
                movlw       0x56
                movwf       vusl
                goto        loadAB

NextLookup17
                movlw       0x1F
                subwf       temperature,w
                btfsc       STATUS,C
                goto        NextLookup18
                movlw       0x57
                movwf       vusl
                goto        loadAB

NextLookup18
                movlw       0x20
                subwf       temperature,w
                btfsc       STATUS,C
                goto        NextLookup19
                movlw       0x57
                movwf       vusl
                goto        loadAB

NextLookup19
                movlw       0x21
                subwf       temperature,w
                btfsc       STATUS,C
                movlw       0x57
                movlw       0x57
                movwf       vusl
                goto        loadAB
```

```
;------------------------------------------------------------------------
;  Binary Floating Point Multiplication :
;  ACCb(16 bits)EXP(b) * ACCa(16 bits)EXPa -> ACCb(16 bits)EXPb
;
F_mpy
            call        setup
mloop       bcf         STATUS,C   ; clear carry bit      ??????????
            rrf         ACCdHI, F     ;rotate d right, place result in W
            rrf         ACCdLO, F
            btfsc       STATUS,C   ;need to add?
            call        D_add
            rrf         ACCbHI, F
            rrf         ACCbLO, F
            rrf         ACCcHI, F
            rrf         ACCcLO, F
            decfsz      temp, F       ;loop until all bits checked
            goto        mloop
            goto        Convert_D
;
setup   movlw  .16         ; for 16 shifts
            movwf       temp
            movf        ACCbHI,W         ;move ACCb to ACCd
            movwf       ACCdHI
            movf        ACCbLO,W
            movwf       ACCdLO
            clrf        ACCbHI
            clrf        ACCbLO        ; clear ACCb ( ACCbLO & ACCbHI )
            return

D_add
            movf        ACCaLO,W        ; Addition ( ACCb + ACCa -> ACCb )
            addwf       ACCbLO, F      ;add lsb
            btfsc       STATUS,C   ;add in carry
            incf        ACCbHI, F
            movf        ACCaHI,W
            addwf       ACCbHI,F      ;add msb
            btfsc       STATUS,C;add in carry
            incf        ACCbHI,F
            return
```

```
;----------------------------------------------------------------
; ReadADC
;----------------------------------------------------------------
ReadADC
            Bank0
            bsf         ADCON0,GO
            btfsc       ADCON0,GO ;loop until conversion is complete
            goto        $-1
            nop
            movf        ADRESH,W
            movwf       temperature
            bsf         ADCON0,GO
            return
;----------------------------------------------------------------
;----------------------------------------------------------------

Convert_D
            movf        ACCbHI,W
            movwf       AARGB0
            movf        ACCbLO,W
            movwf       AARGB1
            movf        ACCcHI,W
            movwf       AARGB2
            movf        ACCcLO,W
            movwf       AARGB3
            call        int_ascii ;convert a 32-bit int to ASCII
            goto        Transmit
;--------------------------------------------------------------------
;--------------------------------------------------------------------
int_ascii
            movlw       last_digit
            movwf       FSR ;pointer = address of smallest digit
            movlw       SIG_FIG ;load counter with the number of
            movwf       digit_count ;significant figures the decimal number
;--------------------------------------------------------------------
flo_asclp
            clrf        BARGB0 ;Make the divisor 10.
            movlw       .10
            movwf       BARGB1
            call        FXD3216U ;divide (32-bit fixed) / 10 (to get remainder)
            movf        REMB1,w ;put remainder in w register
```

```
movwf       INDF ;put number into appropriate digit position
            movlw       0x30
            addwf       INDF,f ;add 30h to decimal number to convert to ASCII
            decf        FSR,f ;move pointer to next digit
            decfsz      digit_count,f
            goto        flo_asclp
            return
            include     <fxd26.a16> ;fixed point 32/16 divide from AN617
            include     <FXM66.a16> ;32 bit float routines
            include      <FP32.a16>
            ;we are using FPM32 for 32-bit multiply
            ;and INT3232 for 32-bit float to 32-bit int
            ;conversion. Routines are in AN575
            return
;--------------------------------------------------------------------------
Transmit              ;send distance computation results to host PC

            movf        hundredthousands,W
            call        Send
            movf        tenthousands,W
            call        Send
            movf        thousands,W
            call        Send
            movf        hundreds,W
            call        Send
            movlw       0x0D
            call        Send
            goto        comp_done
;-----------------------------------------------------------------------------
Send
        Bank1
            btfss   TXSTA,TRMT ; Wait until can send
            goto    $-1
        Bank0
            movwf       TXREG ; Send data in W
            return
;-----------------------------------------------------------------------------
```

```
EXIT_INT                        ; routine to end interrupt
            Bank0               ;select bank 0
            movf      FSR_TEMP,W       ;get saved FSR value
            movwf     FSR         ;restore FSR
            movf      PCLATH_TEMP,W    ;get saved PCLATH value
            movwf     PCLATH           ;restore PCLATH
            movf      STATUS_TEMP,W    ;get saved STATUS value
            movwf     STATUS           ;restore STATUS
            swapf     WREG_TEMP,F      ;prepare WREG to be restored
            swapf     WREG_TEMP,W      ;restore   WREG   without   affecting
STATUS
            retfie              ;return from interrupt
;-----------------------------------------------------------------------
main

Initializations
            Bank0
            CLRF      INTCON
            CLRF      PIR1 ; Clear peripheral interrupts Flags
            CLRF      T1CON ; Stop Timer1, Internal Clock Source,
                      ; T1 oscillator disabled, prescaler = 1:1
            CLRF      TMR1H ; Clear Timer1 High byte register
            CLRF      TMR1L ; Clear Timer1 Low byte register
            CLRF      INTCON ; Disable interrupts
            Bank1
            CLRF      PIE1 ; Disable peripheral interrupts
            BSF       TRISC,0
            Bank0
            MOVLW     0x2A ; External Clock source with oscillator
            MOVWF     T1CON ; circuitry, 1:4 prescaler, Clock source
                      ; is synchronous to device
                      ; Timer1 is stopped
            BSF       T1CON, TMR1ON ; Timer1 starts to increment
                      ;
                      ;  The  Timer1  interrupt  is  disabled,  do  polling  on  the
overflow bit
                      ;

            bcf       error_check,free      ;clear the error_check bit

;--------------------------------------------------------------------
```

```
Loop
          Bank0
          CLRF      INTCON
          CLRF      PIR1
          BSF       INTCON,GIE ; Enable Interrupts
          BSF       INTCON,INTEDG;enable triggering at rising edge of
signal
          BSF       INTCON,INTE;enable external interrupt at pin B0
          BSF       PIE1,SSPIE;enable ssp interrupt
          BSF       INTCON,RBIE
          Bank1
          movlw     SPBRG_VAL ;set baud rate
          movwf     SPBRG
          movlw     0x24          ;enable transmission and high baud rate
          movwf     TXSTA
          Bank0          ;select bank0
          movlw     0x90          ;enable serial port and reception, no errors
          movwf     RCSTA
          Bank1
          BSF       PIE1,RCIE ; Enable receive interrupts
          BSF       PIE1,TMR1IE ; Enable TMR1 Interrupt
;--------------------------------------------------------------------
; InitADC : Analog-Digital Conversion initializations
;--------------------------------------------------------------------
InitADC
          Bank1
          clrf      ADCON1
          bsf       PIE1,ADIE
          Bank0
          movlw     0x81          ;Analog channel is RA0
          movwf     ADCON0             ;ADC is on
          bcf       PIR1,ADIF
          goto      Loop

          END          ; directive 'end of program'
```

Bibliography

[1] Massachusetts Institute of Technology, "The Cricket Indoor Location System," Boston, MA, 2005.

[2] Priyantha B. Nissanka, "Te Cricket Location-Support System", in the 6th ACM International Conference on Mobile Computing and Networking, Boston, MA, 2000.

[3] Hazas, Mike & Ward, Andy "A Novel Broadband Ultrasonic Location System," in *Proceedings of UbiComp 2002: Fourth International Conference on Ubiquitous Computing*, Springer-Verlag, Sweden, 2002.

[5] Want, Roy et al, "The Active Badge Location System," Cambridge, England.

[6] Ward, Andy et al, "The Bat Ultrasonic Location System." Cambridge, UK, 2004.

[7] Bahl, Paramvir & Padmanabhan, N. Venkata, "RADAR: An In-Building RF-based User Location and Tracking System" in *Microsoft Research.*

[8 Welch, Greg et al, "The HiBall Tracking System," in *High-Performance Wide-Area Optical Tracking*, Chapel Hill, NC, 2001.

[9] Randell, Cliff & Muller Henk, "Low Cost Indoor Positioning System," Bristol, UK 2003.

[10] Youssef, Moustafa & Agrawala, Ashok, "The Horus WLAN Location Determination System," College Park, MD.

[11] Wallbaum, Michael & Diepolder, Stefan, "Benchmarking Wireless LAN Location Systems," in *Proceedings of the 2005 Second IEEE International Workshop on Mobile Commerce and Services*, 2005

[12] Hightower, Jeffrey & Borriello, Gaetano, "Location Systems for Ubiquitous Computing," in *IEEE Computer*, August 2001.

[13] www.ubisense.net/Product/whitepapers&downloads.html

[14] Curry, M et al, "Indoor angle of arrival using wide-band frequency diversity with experimental results and EM propagation modeling" in IEEE-APS Conference on Antennas and Propagation for Wireless Communications, 6-8 Nov, 2000, pp 65-68

[15] Fischer, S., & Kangas, A., "Time-arrival estimation for E-OTD location in GERAN" in IEEE International Symposium on Personal, Indoor and Mobile Radion Communications, Sep/Oct 2001, pp F-121-F-125.

[16] Baker, Bonnie, "Temperature Sensing Technologies," Microchip Technology Inc., 1991.

[17] Baker, Bonnie, "Thermistors in Single Supply Temperature Sensing Circuits," Microchip Technology Inc., 1999.

[18] Evans, Rick, "Floating Point to ASCII Conversion," Microchip Technology Inc., 2001.

[19] Palmer, Mark, "Using the PORTB Interrupt on Change as an External Interrupt," Microchip Technology Inc., 1997.

[20] Garbutt, Mike, "Asynchronous Communications with the PICmicro USART," Microchip Technology Inc., 2003.

[21] Schreiber, Robert, "Implementing Ultrasonic Ranging," Microchip Technology Inc., 1997.

[22] Palmer, Mark, "Using Timer1 in Asynchronous Clock Mode," Microchip Technology Inc., 1997.

[23] Dietz, Ken, "Interfacing SPI Serial EEPROMs to PIC16 Devices," Microchip Technology Inc., 2005.

[24] Microchip Technology Inc., "Software Interrupt Techniques," 1997.

[25] Microchip Technology Inc., "EEPROM Memory Programming Specification".

[26] Microchip Technology Inc., "Hardware Techniques for PICMicro Microcontrollers".

[27] Predko Myke, Programming and Customizing PICmicro Microcontrollers, McGraw-Hill Companies, Inc., USA, 2001.

[28] Testa J., Floating Point Math Functions, Microchip Technology Inc., 1997.

[29] http://www.glenbrook.k12.il.us/gbssci/phys/Class/sound/u11l2c.html

[30] Kobitone Audio Company, "Ultrasonic Transducers" datasheet, 05/11/06

[31] Zetex, "ZXM61P03 datasheet", July, 1999

[32] Vishay Semiconductors, "IR Receiver Modules for Remote Control Systems", 08/08/2007

[33] Vishay Semiconductors, "Infrared Transceiver Module (SIR, 115.2 kbits/s) for IrDA applications", 01/26/2004

[34] Microchip Technology Inc., "PICmicro Mid-Range MCU Family Reference Manual", December, 1997

[35] Microchip Technology Inc., "PIC16F87X Datasheet", 2001

[36] www.st.com, July, 2005

[37] Fairchild Semiconductor, "DM74LS00 Quad 2-Input NAND Gate Datasheet", March, 2000

[38] MAXIM, "Dual 256-Tap, Low-Drift, digital Potentiometers in 10 uMAX", 04/2001

[39] MAXIM, "SPI/Microwire-Compatible UART in QSOP-16", 12/2001

[40] MAXIM, "Dual-Output Charge Pump with Shutdown", 03/1996

[41] MAXIM, "MAX232 Datasheet", March, 2004

[42] National Semiconductor, "LMV822 Dual op amp datasheet", December, 2000

[43] National Semiconductor, "LMC7215 datasheet", August, 1996

[44] National Semiconductor, "LM34 Precision Fahrenheit Temperature Sensors datasheet", November 2000

[45] ECS Inc. International, "1.8432 quartz crystal datasheet".